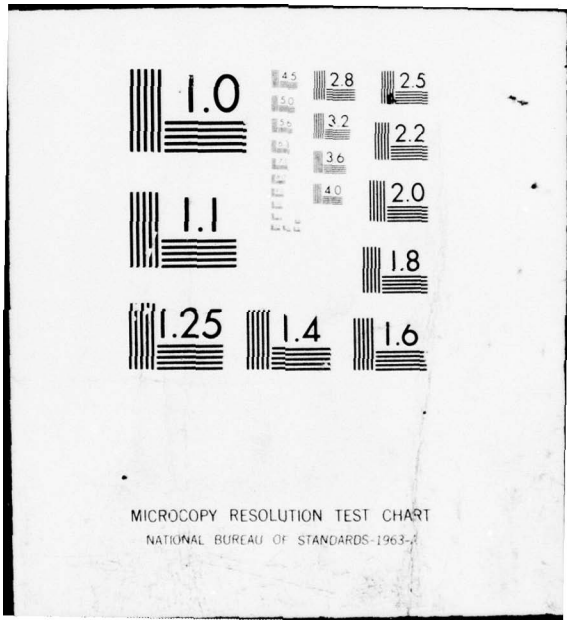
AD-A052 087 ROYAL AIRCRAFT ESTABLISHMENT FARNBOROUGH (ENGLAND) F/G 9/2
USING BRITISH STANDARD INTERFACES TO CREATE A FAST LINK BETWEEN--ETC(U)
JUL 77 G SHARPLES, G F DRURY

UNCLASSIFIED RAE-TR-77010 DRIC-BR-59871 NL

| OF |
AD
A052 087

END
DATE
FILMED
5-78
DDC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

ROYAL AIRCRAFT ESTABLISHMENT

*

(9) Technical Report 77010

(11) July 1977 (12) 56p.

(6) USING BRITISH STANDARD INTERFACES TO CREATE A FAST LINK BETWEEN TWO COMPUTERS

by

(10) G./Sharples
G.F./Drury

*

Procurement Executive, Ministry of Defence
Farnborough, Hants

UDC 681.34 : 681.3.023

ROYAL AIRCRAFT ESTABLISHMENT

Technical Report 77010

Received for printing 20 July 1977

USING BRITISH STANDARD INTERFACES TO CREATE A FAST LINK BETWEEN TWO COMPUTERS

by

G. Sharples

G. F. Drury

SUMMARY

An ICL 1906S running under the *GEORGE 4 operating system* and a Digital Equipment PDP 11/40 have been linked using British Standard Interfaces. The PDP 11 user is given access to the 1906S basic peripherals and filestore via the multi-access facilities of GEORGE. He also has the capability of transmitting files between the filestores of the two machines.

This Report describes the implementation of the inter-machine link giving specifications for the hardware and software involved.

Departmental Reference: Math 227

LIST OF CONTENTS

1 INTRODUCTION

The refreshed graphics installation, provided as a general facility by Computing Division at RAE, consists of a Vector General display linked to a PDP 11/40 minicomputer. The PDP 11 provides the local processing power to run the graphical analysis programs, the core storage from which the picture on the display screen is refreshed and the bulk storage for medium sized data and program files. If the PDP 11 were used in isolation, the maximum size of program which could be run and the limited amount of file storage could prove to be a restriction on the applications which could realistically be undertaken. It was the realisation of these restrictions which led to the development of the system described in this Report.

It was evident that the utility of the graphics installation would be greatly improved if a high speed and readily used means of communication existed between the PDP 11 and the ICL 1906S which provides the central, multi-access computing service for the RAE. It was therefore decided to try to link the two machines. The objective was to create a link with the following characteristics:

(a) the ability to transfer files between the filestore of the 1906S and the filestore of the PDP 11,

(b) a high transfer rate so that a user can transfer files of a reasonable size in an 'acceptable' time,

(c) a simple command language for the user to initiate transfers.

As considerable effort had already been put into connecting a Tektronix display to the 1906S through an ICL 'interface converter'*, or 'uniplexer', this was the obvious means of connection.

It was appreciated at an early stage that the two interfaces would not be immediately compatible and so the precaution was taken of constructing a test box or BSI simulator which could be used to converse with either interface. This would allow major incompatibility problems to be located precisely.

This Report describes the development of the inter-machine link and includes sections on the BSI communications conventions, the hardware of the simulator

---

* An interface converter is a piece of hardware which converts signals conforming to a British Standard[1] into a form suitable for the computer to which it is connected. In order to set up the link two interface converters were required: one produced by ICL and the other by Digital Equipment. Throughout this report these interface converters are referred to as British Standard Interfaces (BSIs).

010

and the software for both the PDP and 1906S computers. Full hardware and software specifications are given in Appendices D and E.

2 A DESCRIPTION OF THE HARDWARE

The equipment used to make the connection is an ICL 7203 Interface Converter unit connected to the 1906S computer, and a BSI interface, produced by the manufacturer of the PDP 11, connected to the PDP 11 input/output highway, (the unibus). This equipment provides a single channel input/output connection between the PDP 11 and the 1906S computer in accordance with British Standard specification 4421 [1].

The ICL 7203 Interface Converter unit transposes signals presented at the BSI into those required for operation of the ICL 1900 *standard interface* and vice-versa. Similarly the Digital Equipment BSI transposes signals presented at the BSI into those required for operation of the PDP 11 unibus and vice-versa.

2.1 Earthing problems

Even assuming compatible interfaces, the two machines could not be linked immediately because of restrictions imposed by ICL on 1906S peripherals. The ICL 1906S computer uses emitter coupled logic in its construction, and the logic elements are switched from the '0' to '1' state, and vice versa, by small voltage changes of the order of 0.9 V on the signal lines. It is therefore essential that the zero voltage reference lines of all peripheral devices and subsystems connected to a 1906S computer are connected directly to the 1906S computer zero voltage point. This will avoid any 'earth loops' which might cause differences in earth potentials in the 1906S computer system which could cause spurious switching of the logic.

The PDP 11 system was originally powered from 13A ring-main wall-sockets. An investigation of the PDP 11 earthing system showed that the PDP 11 zero voltage reference and mains earth were common and that it was not feasible to isolate the two. This meant that if the PDP 11 was connected to the ICL 7203 the 1906S zero voltage reference would be connected to mains earth via the BSI cable and the PDP 11, resulting in an 'earth loop' which could be detrimental to the performance of the ICL 1906S computer.

The problem was solved by powering the PDP 11 system directly from the ICL 1906S generator via a Power Distribution unit.

a Tektronix storage tube to the 1906S [5]. The low level software to handle these links was developed within the Computing Division.

5 DEVELOPMENT OF THE SOFTWARE

The specification of the communications software required that the user be allowed:

(a) To transfer a PDP alphanumeric file into any 1906S file, creating the 1906S file if it did not already exist.

(b) To transfer a PDP alphanumeric file onto any 1906S output peripheral *eg* line printer, card punch.

(c) To transfer a 1906S basic peripheral file into a PDP file.

(d) To manipulate GEORGE files in the usual multi-access (MOP) fashion.

The development was planned to follow several stages:

(a) The transfer of fixed length, fixed content records between the 1906S and the test box.

(b) Linking the PDP 11 to the test box in a similar fashion.

(c) The transfer of fixed length, fixed content records between the two machines.

(d) The transfer of variable length records between the two machines, using the PDP 11 teletype as though it were a normal interactive console linked to GEORGE.

(e) The implementation of the file transfer mechanism.

Because of previous experience in programming for the BSI on the ICL machine, it took only a short time to write a test program for the 1906S which would transfer data into and out of the simulator. Within an afternoon, the closed loop was operational and the test box and 1906S software had run successfully at a high speed transferring a large volume of data.

No corresponding expertise was available for the PDP half of the link and the authors had to rely on the manual [6] supplied with the BSI hardware and a minimum of expertise in PDP assembler programming. The simulator proved to be an invaluable aid and must have reduced the time taken by a considerable factor. Wherever the PDP program failed to communicate correctly with the simulator, the current status of both the BSIs could easily be read from the lamp display, and

the PDP program could be investigated in the usual fashion using the excellent PDP debugging aids for assembler programs.

Having established that the hardware and software at both ends were functioning correctly in isolation, *ie* linked to the simulator, it only remained to complete the direct connection between the two machines. In theory, there should have been no further problems but the two machines adamantly refused to communicate. The resolution of the next series of problems proved to be rather difficult because:

(a) Since both ends worked correctly at high speed with the simulator, neither end could be identified as the 'trouble maker'.

(b) Without the simulator in the circuit, it was only possible to establish the status of the ICL BSI by investigating the depths of the hardware with an oscilloscope probe and consulting the corresponding circuit diagram.

(c) The ICL BSI (or GEORGE Executive) demonstrated again its lack of resilience, as it had during its use as a link to the Tektronix display. The BSI would periodically go into an indeterminate 'lock out' state, in which it rejected all requests from the 1906S and all operations proposed by the remote device. Sometimes, this condition could be reset by switching the BSI off and then on again after a pause of a couple of minutes; but at other times testing would result in the total loss of input and output on the 1906S, which was understandably unpopular both with users and with the operations staff. System tests had, therefore, to be made with care.

(d) The failures did not occur consistently but appeared to vary according to the work load being processed by GEORGE

After a period of intensive development, using mainly a trial and error approach, the two machines were able to transfer fixed length, data blocks to each other and the authors were able to progress to the more interesting problem of providing the interface software to allow the graphics user to transfer and manipulate his 1906S and PDP 11 files.

The software already in use with the ICL BSI has been described elsewhere[4]. Its purpose is to allow a remote device linked to the BSI to function as if it were a normal MOP console but with the additional advantages of a potentially high speed of working, plus the ability to communicate in terms of the full ASCII character set. The software is known as a driver since it handles the transfer of data between two normally unconnected input/output channels: 010

(a) it controls the data flow to and from the BSI,

(b) using the command issuer facility of GEORGE, it issues commands to GEORGE and receives replies, as if it were a normal multi-access user of the system.

In normal operation, the driver accepts the data arriving at the BSI, passes the completed data buffer to GEORGE, reads the reply from GEORGE and then writes this back to the BSI. Thus the remote device functions in a conversational mode. Using this software unmodified, it appeared that all the requirements could be satisfied. At the start of a file transfer or manipulation session the PDP 11, the user establishes the link to the 1906S which gives him the full MOP capability. He can then LOGIN to GEORGE and the transfer operations at the 1906S end can be requested using the normal GEORGE commands:

(a) 1906S files can be created or overwritten using an INPUT command.

(b) Information can be transferred onto an output peripheral using an INPUT command followed by a LISTFILE. These commands would be generated automatically by the PDP 11 driver in response to a file transfer request from the user.

(c) 1906S basic files can be output using a LISTFILE command.

(d) 1906S files can be manipulated in the normal MOP fashion.

The advantage of this proposal was that it required programming effort only at one end of the link, *ie* a PDP driver had to be written to pass commands from the keyboard to the 1906S and to read back the replies. Special pseudo GEORGE commands requesting file transfers would be trapped by the driver, which could pass across the required GEORGE command and then handle the subsequent file transfer.

Early in the development of the PDP driver it became apparent that this system had some major drawbacks:

(a) Because of the work-load on the 1906S service from other users, it was not always possible to LOGIN, the system being 'full'. The file transfer facility was thus effectively not available.

(b) On transferring data to the 1906S, the PDP 11 program had to wait for GEORGE to respond between each record. When using an INPUT command, the procedure is for GEORGE to say 'ready', the user presents his data, and when GEORGE is prepared to accept the next line, it outputs the 'ready' signal again.

This conversational mode necessarily slowed down the transfer process, especially during periods when the 1906S was heavily loaded.

It thus became necessary to redesign the system with the objectives of allowing the graphics user to transfer files without having to LOGIN to the 1906S, and secondly, to increase the transfer rate. A representation of the original design is given in Fig 2.

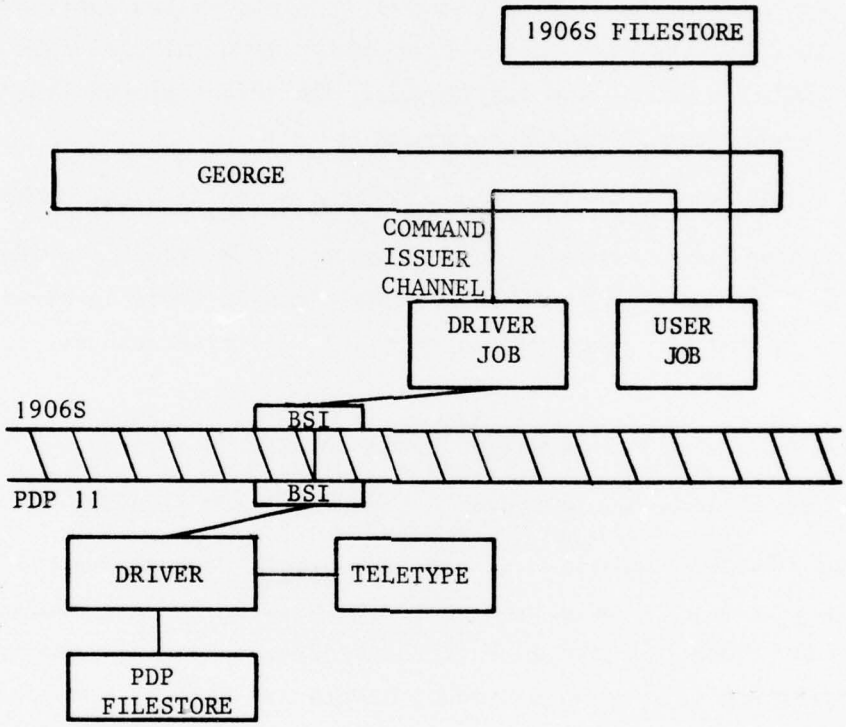


Fig 2 A representation of the original design for the file transfer system

From this diagram, it can be seen that one link in the chain is removed if the driver job itself executes the 1906S filestore transfers rather than setting up another job on behalf of the user when the user's LOGIN command is passed to GEORGE. The ICL BSI driver does not have the same problem of 'logging in' since this job has the privileged 'system issued' status, which means that it is initiated by the operators and becomes started immediately. This solves the problem of the user not being able to initiate his own job but it also introduces additional complications since the driver is not run under the user's own cost code. The normal filestore access restrictions apply and thus it is not possible for the graphics user to transfer to or from a 1906S file which has not been 010

appropriately trapped (see section 6: User Specification) and it is not possible to create files in the user's directory by means of a file transfer request. This latter drawback has been reduced in importance by setting up a central directory, controlled by the cost code under which the driver operates, in which the graphics user can create or overwrite files. Unless the default is explicitly overridden all 1906S files referred to in a filestore request are assumed to belong to the directory :MM1132-PDP, which is an inferior to the proper user :MM1132. If the user does successfully LOGIN to the 1906S then he can trap or create files under his own directory before he requests the file transfer.

The second problem concerning the response to an INPUT command is avoided by the driver issuing direct input/output requests to the user's file which has been assigned to the driver. This is to be contrasted with the former method whereby information was transferred in a (slow) conversational fashion through the command issuer channel.

The cost of the two improvements appears in the form of greater complication of use and in the effort in enhancing the 1906S simulator to recognise and implement file transfer requests from the PDP 11.

Even in this final stage of development, problems occurred in transferring information from one machine to the other. These problems appeared to be mainly associated with the transfer of short length records so that the BSIs had to change very quickly between read and write status. The 1906S software had previously only handled a device which responded relatively slowly and so some modifications were made to speed up the driver program. Also, as an additional safeguard, the PDP routines were written to accept a slave relationship to the 1906S, monitoring the status of the ICL BSI and accepting this as correct. Thus should the two drivers get out of step, the PDP 11 driver would fall into line with the current request set up by the 1906S driver, and the possibility of 'locking up' the 1906S is avoided.

6 USER INTERFACE TO THE SOFTWARE

In order to establish the connection between the PDP 11 user and the 1906S, it is necessary to initiate a driver program in each machine. The driver in the PDP 11 must be started first. This sends an introductory message to the teletype:

PDP 11-1906S LINK

010 and waits for a message from the 1906S. The 1906S driver can then be started

up and first of all, it transfers the GEORGE 'hello' message to the PDP 11 driver, which in turn transfers it to the teletype:

RAE MOP SERVICE

The PDP 11 user is then faced by an almost normal MOP situation and after logging in he is free to manipulate his 1906S files using the normal GEORGE commands from the PDP 11 teletype.

The PDP 11 driver examines all commands received from the user to establish if he wishes to execute a file transfer. This the user indicates by typing one of the special commands

CYI - copy file in
CYO - copy file out

Two parameters to these commands specify the PDP 11 and 1906S files or peripherals to be used in the transfer. On receipt of one of these commands, the PDP 11 driver inhibits the transfer of the command to the 1906S but instead sends across a data header indicating the 1906S file or peripheral and the direction of the transfer.

For an inward transfer, the PDP 11 driver then reads a series of records from the 1906S storing them in the *PDP 11 file until a transfer* termination record is received. Conversely, for an outward transfer, it reads a series of records from the PDP 11 file and transfers them across to the waiting 1906S. On recognition of the end of the PDP 11 file, a transfer termination record is transmitted to the 1906S.

On the completion of the transfer, the driver reverts to the MOP mode, transferring commands direct to the 1906S, until another file transfer request is recognised.

A further special command is issued by the user to close down both drivers:

RELEASEM.

The 1906S driver acknowledges receipt of this command by transmitting to the PDP 11 the message:

CLOSED DOWN

and it then deletes itself. The PDP 11 driver transfers this message to the teletype and similarly deletes itself, returning control to the operating system. In this state, the PDP 11 user is isolated from the 1906S and any further commands input are handled by the PDP operating system.

Further details are given in Appendix C.

7 THE SPEED OF THE LINK

Some elementary timings measurements were made in order to obtain a feeling for the speed of the link, its potential speed and perhaps to identify any factors reducing the transfer rate. Table 1 gives a typical timing result obtained for each method of connection, together with a final, rounded, average transfer rate. All transfer rates are expressed in bytes per second, where 1 byte consists of 8 bits.

The reservations about these figures should be stated first:

(a) The readings were necessarily taken over a period of days and so the work load being handled by GEORGE did vary. However, all the figures were taken when the system was moderately loaded, *ie* about 13 MOP jobs and two background jobs, and so they should represent a typical case.

(b) In some cases, the quantity of data transferred was relatively small and so the file initialisation procedures account for a higher proportion of the total time than they would for larger transfers. This has the effect of depressing the figure for the average transfer rate.

(c) Timings were only taken to the nearest second and so where the total transfer time is quite small, significant errors could be involved.

For the first trial, the authors examined the transfer rates between the simulator and each of the two machines in turn. The two sets of results proved to be similar: the transfer rate was 20000 bytes per second out of the simulator and 1500 bytes per second in the opposite direction. The 1500 bytes per second figure is obviously a limitation of the simulator, which was not specifically designed to act at high speed; in fact, delays were deliberately incorporated so that the transfer pattern could be viewed more easily on an oscilloscope. Similarly, the two transfer rates out of the simulator were the same and so it appears that this also is a limitation imposed by the test box. Nevertheless, for this primarily qualitative assessment, 20000 bytes per second seems a sensible maximum operational transfer rate.

When the simulator was removed and the two machines communicated using fixed length, fixed content messages, the transfer rate dropped by a factor of 2 to around 10000 bytes per second. This is presumably caused by the delays introduced with synchronising two independent processes, both performing data
010 initialisation and checking. The use of a filestore system at either end caused

a further reduction in the transfer rate by a factor of 2 or 3. In this respect, it appears that the PDP DOS filing system has the greater effect.

Finally, the use of the complete system, *ie* transferring from one filestore to the other, involved little further reduction in the transfer rate presumably because much of the file handling work in the two machines can be carried out in parallel. However, by this stage, the transfer rate had fallen from the initial figure of 20000 bytes per second to around 2500 bytes per second, a considerable drop.

Even at this reduced speed, the system can still handle bulk quantities of data reasonably quickly: to transfer a complete PDP 11 disk of 2.4 million bytes would take approximately 15 minutes. But the graphics user is unlikely to be handling such large quantities of data since it could never be refreshed on the display screen. A reasonable sized data file might be 100000 bytes and this can be passed across in less than 1 minute.

Table 1

Timings taken to estimate the speed of the link

Direction of transfer	Quantity of data	Time	Average transfer rate
Test box to 1906S	14000 blocks of 256 bytes	165 s	20000 bytes/s
1906S to test box	1100 blocks of 256 bytes	183 s	1500 bytes/s
Test box to PDP 11	1000 blocks of 256 bytes	13 s	20000 bytes/s
PDP 11 to test box	1000 blocks of 256 bytes	165 s	1500 bytes/s
PDP 11 core to 1906S store	1000 blocks of 256 bytes	24 s	10600 bytes/s
1906S core to PDP 11 core	1000 blocks of 256 bytes	21 s	12200 bytes/s
PDP 11 file to 1906S store	100 blocks of 256 bytes	9 s	2800 bytes/s
1906S core to PDP 11 file	500 blocks of 256 bytes	37 s	3400 bytes/s
1906S file to PDP 11 core	100 blocks of 256 bytes	5 s	5100 bytes/s
PDP 11 core to 1906S file	1000 blocks of 256 bytes	44 s	5800 bytes/s
PDP 11 file to 1906S file	100 transfers of 256 bytes	11 s	2400 bytes/s
1906S file to PDP 11 file	100 transfers of 256 bytes	10 s	2600 bytes/s

8 CONCLUSION

The objectives of the project were achieved in that a reliable and high speed link has been created between the 1906S and the PDP 11/40. The link is 010

convenient to use since it operates in terms of the GEORGE command language and information can be transferred directly from filestore to filestore without having to resort to an intermediate medium such as paper tape.

Large and long-running data reduction and analysis programs can be run on the 1906S utilising the processing power and program capacity of a large computer backed up by a large, secure filestore. Only the final stage of data processing need be run on the PDP 11 when the user requires intimate interaction with his experimental results. This means that the PDP 11 is fulfilling its proper role of supporting the graphics display rather than performing number-crunching calculations.

In addition to providing access to the processing power and filestore of the 1906S, the link has also allowed the number of peripherals linked to the PDP 11 to be reduced, with the emphasis being placed on graphical input and output devices rather than the normal paper tape, line printer and card devices. All these facilities are already provided by the 1906S installation and thus a considerable duplication of hardware has been avoided[7]. Again, the graphics installation is pursuing its true goal and the computer hardware has not been allowed to proliferate and assume an undeserved importance.

Appendix A

THE BSI COMMUNICATION CONVENTIONS

A BSI consists of two logical sections: a Source and an Acceptor. Data is transferred from the source of one interface to the acceptor of the other interface and transfers are regulated using control signal lines which indicate 'operable', 'termination' and 'ready to accept transfer' (control) conditions.

A.1 Data Transfers

The BSI Source will set its SOURCE OPERABLE (SO) line when the source computer is ready to transfer data to the acceptor computer. This SO indicates to the BSI Acceptor that the Source is operable and able to send data when required to do so. In response to SO the Acceptor makes itself ready to receive data and, when ready, sets its ACCEPTOR OPERABLE (AO) line, indicating to the Source that it is operable, and its ACCEPTOR CONTROL (AC) line to request the data transfer. SO and AO must both be set before any data transfer can take place.

In response to AO and AC the Source will cause the first data byte to be set onto the 8 BSI data lines and will then set its SOURCE CONTROL (SC) line to indicate to the Acceptor that data is ready. The data byte is then copied from the BSI data lines into the acceptor computer memory via the BSI Acceptor. Upon completion of the transfer the Acceptor resets its AC line to indicate to the Source that the data has been accepted. The Source notes that the data has been accepted and resets its SC line.

As soon as the acceptor computer is ready to receive the next data byte the BSI Acceptor will set its AC line to request the next data transfer from the BSI Source.

A.2 Termination of a data transfer

Data transfers continue in the manner described (section A.1) until the BSI Source sets its SOURCE TERMINATE (ST) line coincident with its SC line, indicating to the BSI Acceptor that the coming data byte is the last in the current block. The transfer of this last data byte will take place in the normal way but the BSI Acceptor, having detected ST, will reset its AO line when its AC line is reset. The BSI Source notes that the data block has been accepted (AO is reset) and resets its SC, SO and ST lines. The BSI is then in a correctly terminated state ready to proceed with the next block transfer.

Appendix B

THE SIMULATOR OPERATING MODES

B.1 Read-write mode

The simulator, when operating in this mode, first acts as an Acceptor and receives (READs) 256 byte blocks of data, in the form of a binary count, from a Remote Source, checks each byte as it is received and indicates if any errors occur by lighting the error lamp.

Upon completion of the READ transfer the simulator automatically disables its Acceptor and enables its Source. It then transfers (WRITEs) a hardware generated 256 byte data block, in the form of a binary count, to a Remote Acceptor where the data can be checked.

The simulator can be made to stop transfers after each READ/WRITE cycle, and then be manually restarted, or can 'free run' alternately issuing READs and WRITEs.

B.2 Read only mode

In this mode the simulator continuously READs and checks data blocks from a Remote Source, indicating any errors.

B.3 Write only mode

In this mode the simulator continuously generates data blocks and transmits (WRITEs) the data to a Remote Acceptor where it can be checked.

B.4 Test mode

The simulator Source must be connected to the simulator Acceptor when operating in this mode. The simulator generates data blocks which are transferred from the Source to the Acceptor where the data is checked. A data error will stop transfers. This mode is used to check that the simulator is functioning correctly.

Appendix C

USER SPECIFICATION OF THE SOFTWARE

C.1 Establishing the link

In order to transfer data in either direction it is necessary to initiate a driver program in each machine. The PDP program should be started first and this is done by:

(a) Logging in as usual.

(b) Typing in: R GEORGE

The program replies with the message:

PDP11-1906S LINK

and waits for a message from the 1906S.

The 1906S driver is activated by informing the operators that the display link is required.

When contact is established the usual

RAE MOP SERVICE

message is output on the teletype.

C.2 Using the link

Having received the 'hello' message the user is faced by an almost normal MOP situation. The differences are:

(a) To transmit a line RETURN is pressed and not ACCEPT or ESCAPE.

(b) To cancel a line, press CONTROL&U. The PDP driver replies with a CARRIAGE RETURN and LINEFEED and the correct line may then be input.

(c) The GEORGE facility to delete a character may not be used in the file transfer commands. To delete a character, press RUBOUT: the driver acknowledges this request with a '←'.

Files may be transferred by using two special commands:

CYI - copy in
CYO - copy out

These are recognised by the PDP driver and are not passed across to the 1906S.

010

The format of the commands is:

CYI PDP dataset specification, 1906S source specification
CYO PDP dataset specification, 1906S destination specification.

The system has been designed so that files can be transferred in no-user context and so the user need not LOGIN to the 1906S. A consequence of this, however, is that the 1906S files are accessed directly by the 1906S driver which is run under a pseudo user, :MM1132-PDP, which is subordinate to the proper user, :MM1132. Thus files to be transferred from the 1906S to the PDP must be given READ traps for :MM1132; and files in the user's directory which are to receive data must already exist and must be given the WRITE trap for :MM1132.

Eg TG OUTFILE, :MM1132,READ - file for transfer to PDP 11
TG INFILE, :MM1132,WRITE - file for transfer from PDP 11.

Users should note that while a 1906S file is trapped to :MM1132, any PDP 11 user can access that file and so it is strongly recommended that WRITE traps be removed at the end of a session.

If no user name is specified for the 1906S file, :MM1132-PDP is assumed.

Eg CYO FRED.LST,FRED-LST

establishes or overwrites a 1906S file, :MM1132-PDP.FRED-LST from the contents of the PDP file, FRED.LST.

Eg CYO DK1:JOE.MAP [1,1],:MM1199.JOEMAP

overwrites the 1906S file, JOEMAP, belonging to :MM1199.

If several files are to be transferred using the same 1906S directory, the name of the default directory (initially :MM1132-PDP) can be changed by using the additional special command, PDPTDY (PDP Transfer DirectorY).

Eg PDPTDY :MM1199.

After entering the command given above, a user name of :MM1199 is assumed for all 1906S files with no explicit user name.

Generation numbers and language codes may be specified for the 1906S files but not qualifiers such as APPEND.

The ICL specification is optional. If none is given, it is assumed that the filename for the 1906S should be the same as that for the PDP machine, any '.'s being replaced by '-'s.

Eg CYO BILL.FTN 010

establishes or over writes a 1906S file, :MM1132-PDP.BILL-FTN.
This file can then be returned to the PDP 11 by a similar command :

CYI BILL.FTN

The ICL specification need not be a filename but can be an output device plus any qualifiers which are allowed in a LISTFILE command:

eg CYO DICK.BAK,*TP
CYO HARRY.MAC,*LP,PR MATHS,FR 100

In these cases, the file is copied into a workfile and the workfile is then listed on the appropriate device.

The PDP 11 file specified in a CYI command must not exist at the time the command is issued.

The maximum size of record which can be transferred in either direction is 256 bytes.

C.3 Errors in file transfers

Two error messages can be produced as the two specifications are checked:

(a) DEC ERR : the program has failed to open the specified PDP file. No information has been passed to the 1906S and the program issues a further request to the keyboard for the next command.

(b) ASSIGN ERR : the PDP file was opened successfully but the 1906S driver has failed in trying to perform an assignment on the specified 1906S file. A request is made to the keyboard for the next command.

If the 1906S driver recognises an error in a course of a transfer it continues until the file transfer is complete and then issues the message:

TRANSFER ERR

It is then up to the user to decide what action to take.

If the PDP driver recognises an error in transferring to or from disc it halts. Pressing the CONTINUE switch causes the error to be ignored.

C.4 Disconnecting the link

The link is disconnected by:

(a) Issuing a LOGOUT command if the user has logged in to GEORGE.

(b) Typing 'RELEASEM'.

The 1906S driver confirms that it has received the message by returning a 'CLOSED DOWN' reply and the PDP 11 then returns to the normal command status. Both the drivers have then been removed.

Appendix D

LOGICAL OPERATION OF THE BSI SIMULATOR

The BSI Simulator logic is shown in Figs D1 and D2 and typical waveforms in Fig D3.

A logic reset signal (REST) is generated whenever the simulator is switched on or the RESET switch is operated.

D.1 Operation in auto mode

D.1.1 Simulator write 'transfer'

The signal REST resets the counter flip flops C0 to C7 and the dual purpose, counter source operable/acceptor operable (CSO/AO). This flag is in fact just a ninth counter flip flop which is alternately set and reset as the count in C0 to C7 increments from 257 to 0. REST causes the simulator to signal SIMULATOR SOURCE OPERABLE (SSO) to the Remote Acceptor indicating its ability to commence data transfers.

The Remote Acceptor notes SSO and, when it is ready to accept data, sets its REMOTE ACCEPTOR OPERABLE (RAO) and REMOTE ACCEPTOR CONTROL (RAC) lines. The simulator responds to RAC by setting its SIMULATOR SOURCE CONTROL (SSC) line, indicating that the first data character (the contents of the counter C0 to C7) is on the data lines SD0 to SD7. The data is then copied from the data lines to the remote computer. On completion of the transfer the Remote Acceptor drops RAC to indicate that the data has been accepted. The simulator responds by dropping SSC. Each time RAC is dropped the counter C0 to C7 is triggered causing its contents to be incremented by one. The Remote Acceptor sets its RAC line once again when it is ready to receive the next data character thereby initiating the next transfer.

The above sequence of events continues until the contents of the counter is equal to 255, *ie* the counter outputs are all binary '1'. The simulator then sets its SIMULATOR SOURCE TERMINATE (SST) line to indicate that the coming character is the last in the current block. This data transfer takes place as before but the Remote Acceptor drops RAO when RAC is dropped. The simulator drops SSC and SST in response to the change in state of RAC. The latch TERMINATE is triggered when SST is dropped and this causes the SSO line to be disabled, resulting in SSO being dropped, and the SIMULATOR ACCEPTOR OPERABLE (SAO) being enabled.

The counter is again triggered when RAC is dropped and this causes the
010 counter outputs to be set to binary '0' and the CSO/AO flip flop to change state.

The signal SAO is therefore set indicating that the simulator is ready to accept data from the Remote Source.

D.1.2 Simulator 'READ' transfer

The Remote Source sets its REMOTE SOURCE OPERABLE (RSO) and REMOTE SOURCE CONTROL (RSC) lines when it is ready to transfer the first data character to the simulator. In response to RSC the simulator sets its SIMULATOR ACCEPTOR CONTROL (SAC) line indicating that it is ready to accept data. The simulator then compares the data on the data input lines (RD0 to RD7) with the contents of the counter and after a short period of time (set by Monostable RESET SAC) drops SAC. The Remote Source notes that the data has been received (SAC) and drops RSC. If an error has been detected in the data SAC is inhibited and transfers cease but if no errors are found SAC is set again to request the next data character. Data checking can be inhibited by means of the DATA CHECK switch.

The counter, C0 to C7, is triggered when RSC is dropped at the end of each data character transfer, causing its contents to be incremented by one each time, and so data from the Remote Source must be in the form of a binary count if data checking is to be performed.

The above sequence of events continues until the data character equals 255, when the Remote Source will set its REMOTE SOURCE TERMINATE (RST) line coincident with RSC, indicating that this is the last transfer of the current block. This last data character is transferred as before but at the end of the transfer, when RSC is dropped, RSO and RST are also dropped. The new state of RSO prevents SAC being set again. Also the latch TERMINATE is triggered when RST is dropped and this causes the SAO line to be inhibited, resulting in SAO being dropped, and the SSO line to be enabled. The counter, C0 to C7 and CSO/AO, is again triggered when RSC is dropped and this causes the contents of the counter to be set to zero and the CSO/AO flip flop to change state. This results in SSO being set again causing the above sequence (from section A.1.1) to be repeated.

Control of the SSO and/or SAO lines by the latch TERMINATE can be inhibited by operation of the SSO and/or SAO CONTROL switches. Also the data transmitted to the Remote Acceptor from the simulator can be set to any fixed bit pattern by means of the SELECT DATA SOURCE switch and data select switches, DSW0 to DSW7. The SELECT DATA SOURCE switch routes either the counter outputs or the data switch outputs onto the simulator data output lines.

D.2 Operation in WRITE mode

The MODE SELECT switch, when set to the WRITE position, disconnects the CSO/AO flip flop outputs from the SSO and SAO lines and permanently holds SSO set and SAO reset. As a result the simulator can only transmit (WRITE) data to a Remote Acceptor as described in section A.1.1. All other control switches operate as before.

D.3 Operation in READ mode

The MODE SELECT switch, when set to the READ position, disconnects the CSO/AO flip flop outputs from the SSO and SAO lines and permanently holds SAO in the set state and SSO in the reset state. As a result the simulator can only receive (READ) data from a Remote Source as described in section A.1.2. All other control switches operate as before.

D.4 Operation in LOOP mode

The MODE SELECT switch, when set to the LOOP position disconnects the CSO/AO flip flop output from the SSO and SAO lines and permanently holds SSO and SAO in the set state. To operate in this mode the Simulator Source interface must be connected to the Simulator Acceptor interface, SSO to RAO, SAO to RSO, SDO to RDO, SD1 to RD1 etc, so that information transmitted from the Simulator Source is received by the Simulator Acceptor. This mode of operation allows the simulator to test itself. All other control switches operate as described above.

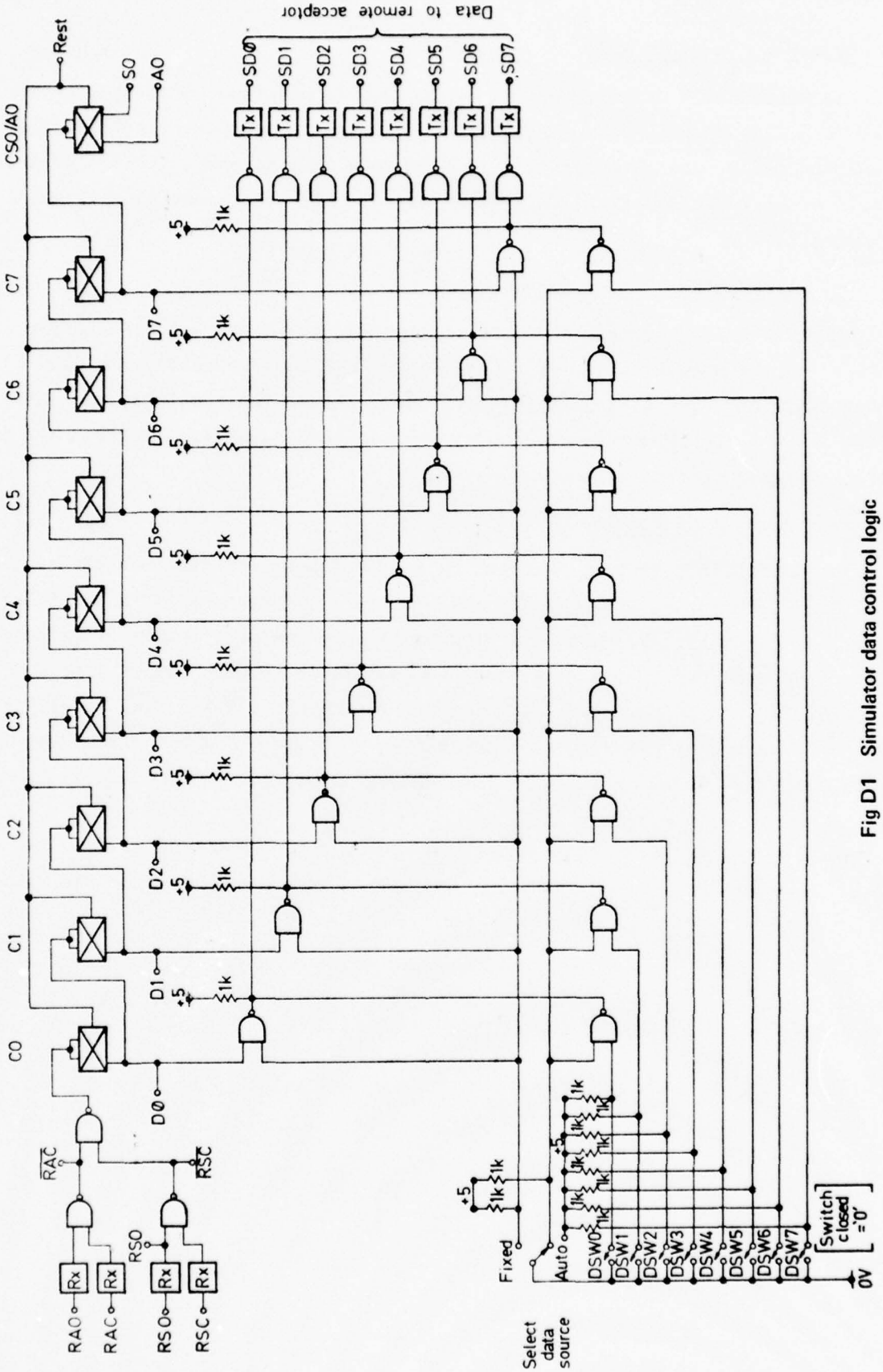


Fig D1 Simulator data control logic

TR 77010

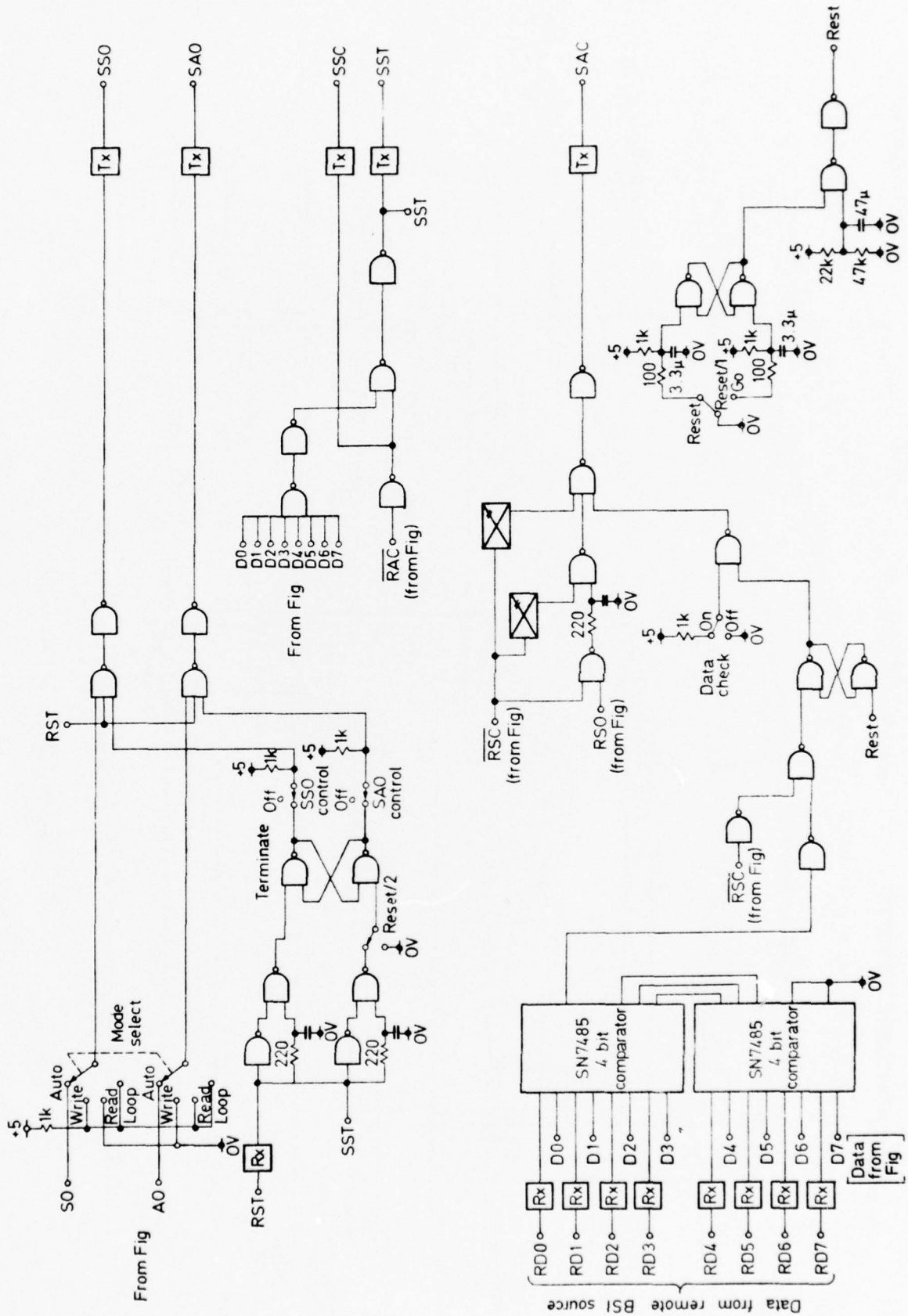


Fig D2 Simulator BSI timing control logic

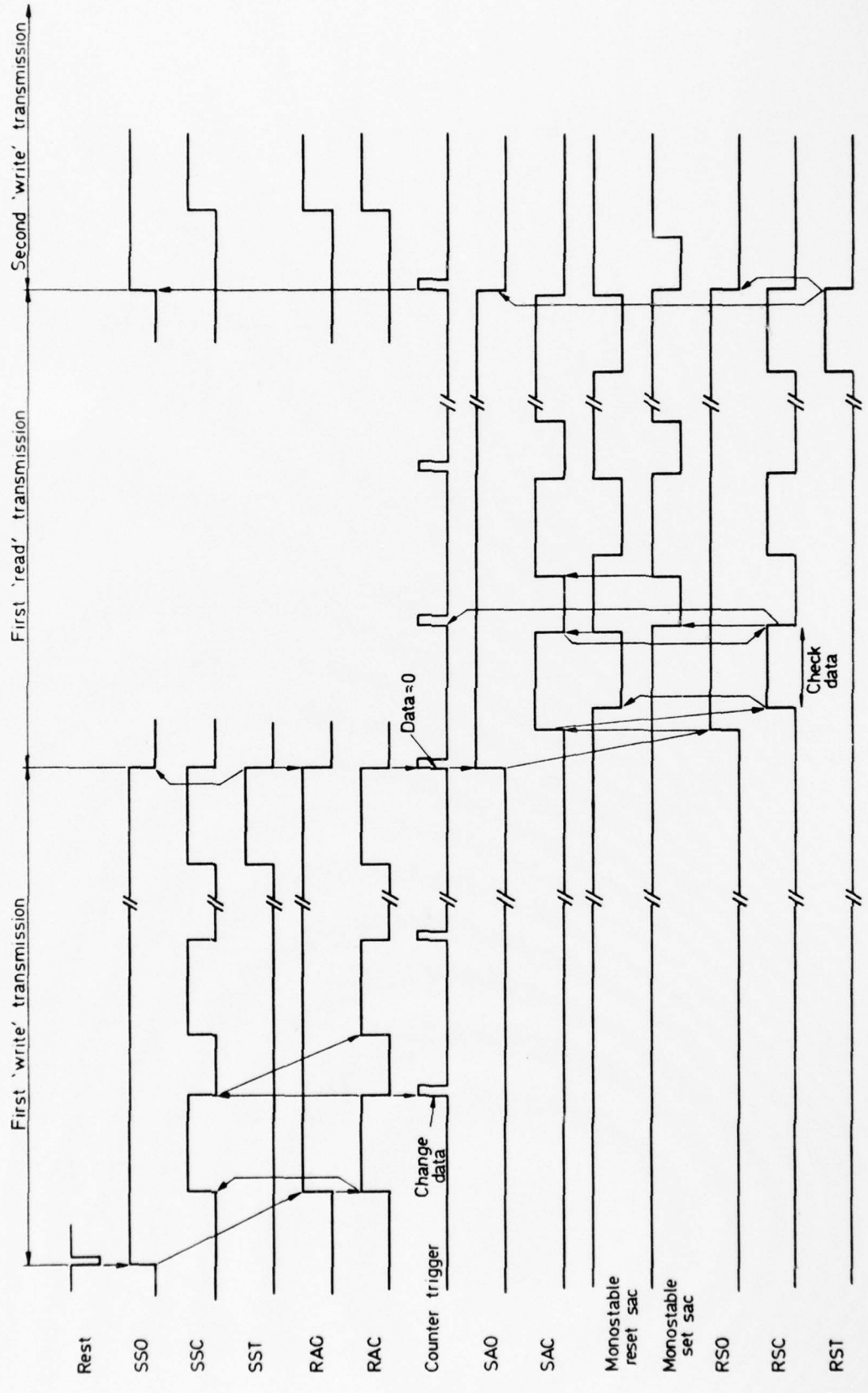


Fig D3 Simulator logic waveforms

Appendix E

SOFTWARE DOCUMENTATION

The software consists of two communicating programs, one running in the PDP 11 machine and the other in the 1906S. This system allows a user at the PDP 11 teletype:

(a) full access to the MOP facilities of GEORGE,

(b) the ability to transfer

(i) 1906S alphanumeric files into PDP 11 files,

(ii) PDP 11 alphanumeric files into 1906S basic files,

(iii) PDP 11 alphanumeric files onto 1906S basic peripherals.

E.1 1906S software description

The original driver to control the ICL BSI has been described in detail elsewhere[4] and so only the modifications required to implement the file transfer procedure are described in this section. Figs E1 to E5 contain the flowcharts for the routines concerned. The driver is written in the ICL assembler language, PLAN.

The main controlling routine implements the loop required to provide the conversational MOP facility, *ie*

(a) read a command from the BSI and pass it to GEORGE,

(b) read the reply from GEORGE,

(c) pass the reply to the BSI,

(d) if GEORGE is ready to receive a further command, return to step (a); otherwise step (b).

Step (a), as described above, is handled by the subroutine MINT. It inputs the command typed on the PDP 11 teletype and passes control to the routine CHIN, which checks if the request:

(a) is to be passed to GEORGE for processing, or,

(b) is to be processed by the driver itself.

CHIN has two exits corresponding to these two conditions. Four 'non-GEORGE' requests are trapped for internal processing:

(a) RELEASEM : the user wishes to terminate his use of the BSI link and thus the driver program is deleted with the message 'OK'.

(b) PDPTRANO : the PDP 11 user wishes to transfer a file to the 1906S and to implement this request, the routine TROUT is entered.

(c) PDPTRANI : the PDP 11 user wishes to transfer a file from the 1906S and to implement this request, the routine TROP is entered.

(d) PDPTDY : the PDP 11 user wishes to change the default directory for file transfers.

The 1906S destination of a file being transferred from the PDP 11 is passed to the 1906S in the 'PDPTRANO' warning record. Where the specification is a file-name then the output stream can be directed to that file in a straightforward manner. If a device is specified (*ie* the first character of the specification is a '*' requesting a basic device such as a line printer, *LP) then the data stream must be directed to a workfile, which is listed on the required device and then erased. In fact, the LISTFILE and ERASE commands are issued immediately after the data stream has been assigned but they are not implemented until the transfer is complete and the file is released. Should a command error occur during the execution of any of these commands than a special exit from TROUT is taken to indicate that the transfer cannot be undertaken and that an error message should be output to the BSI : 'ASSIGN ERR'. If the data stream is assigned successfully, then all subsequent data read from the BSI is transferred to the file until the PDP declares that the 'end of file' has been reached by sending the message 'PDPTRANE'. On exit from TROUT, the driver returns to MINT to take the next command from the BSI.

Similarly, for a file transfer to the PDP 11 the 1906S file specification is contained in the 'PDPTRANI' record sent to the 1906S and the routine TROP assigns the input stream appropriately. Should a GEORGE command error occur then a special exit is taken, exactly as for TROUT; otherwise the whole of the file is transferred to the PDP 11 and the 'end of file' is announced by sending the terminating record, 'PDPTRANE'. On exit, the driver loops to accept the next command typed on the PDP 11 teletype.

Should any BSI operation generate an unexpected reply word then at the end of the file transfer the message 'TRANSFER ERR' is sent to the PDP teletype as a warning to the user that his data is suspect.

To change the default directory, the driver simply issues a DIRECTORY (DY) command with the requested directory name as a parameter.

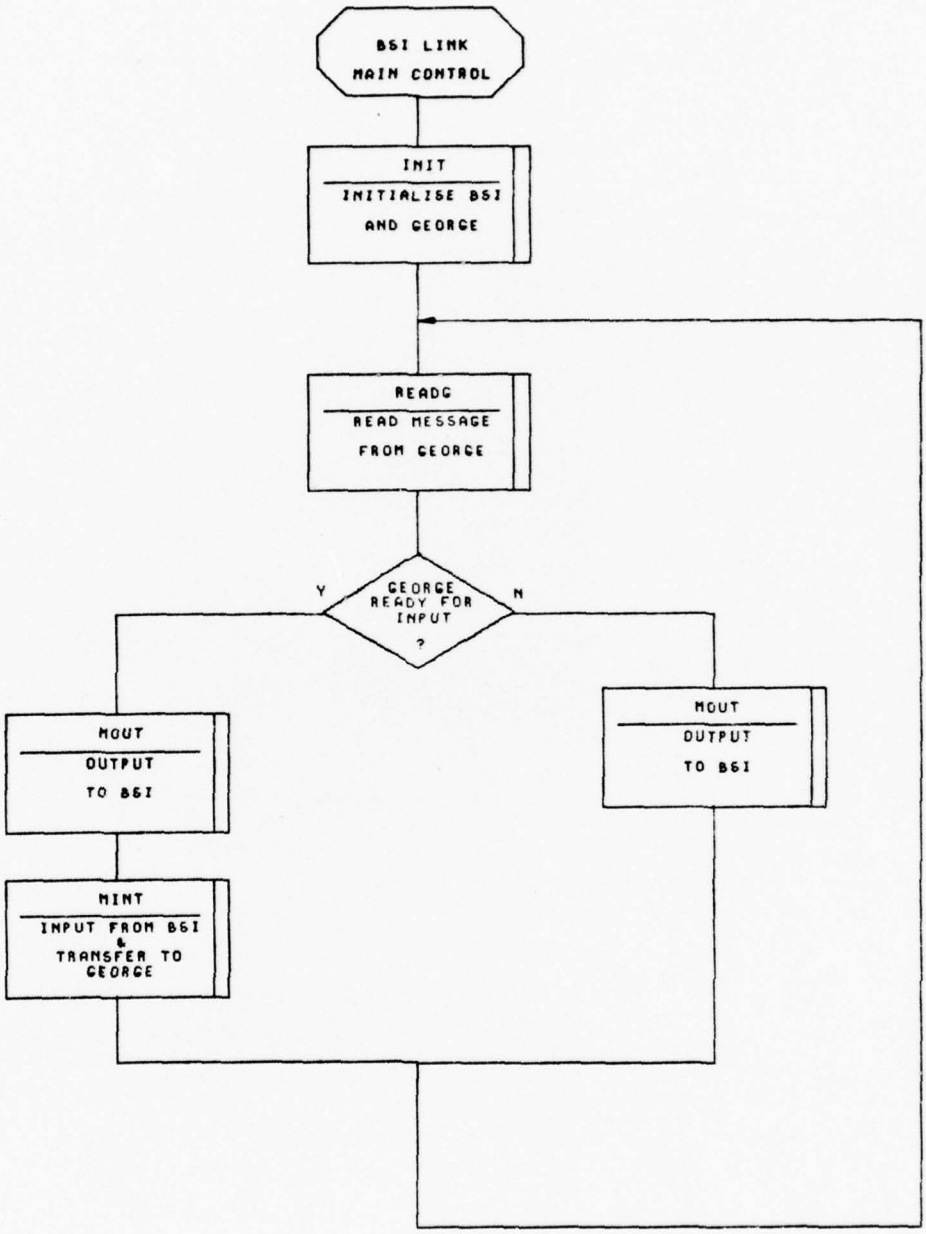


Fig E1 BSI link: 1906S main control

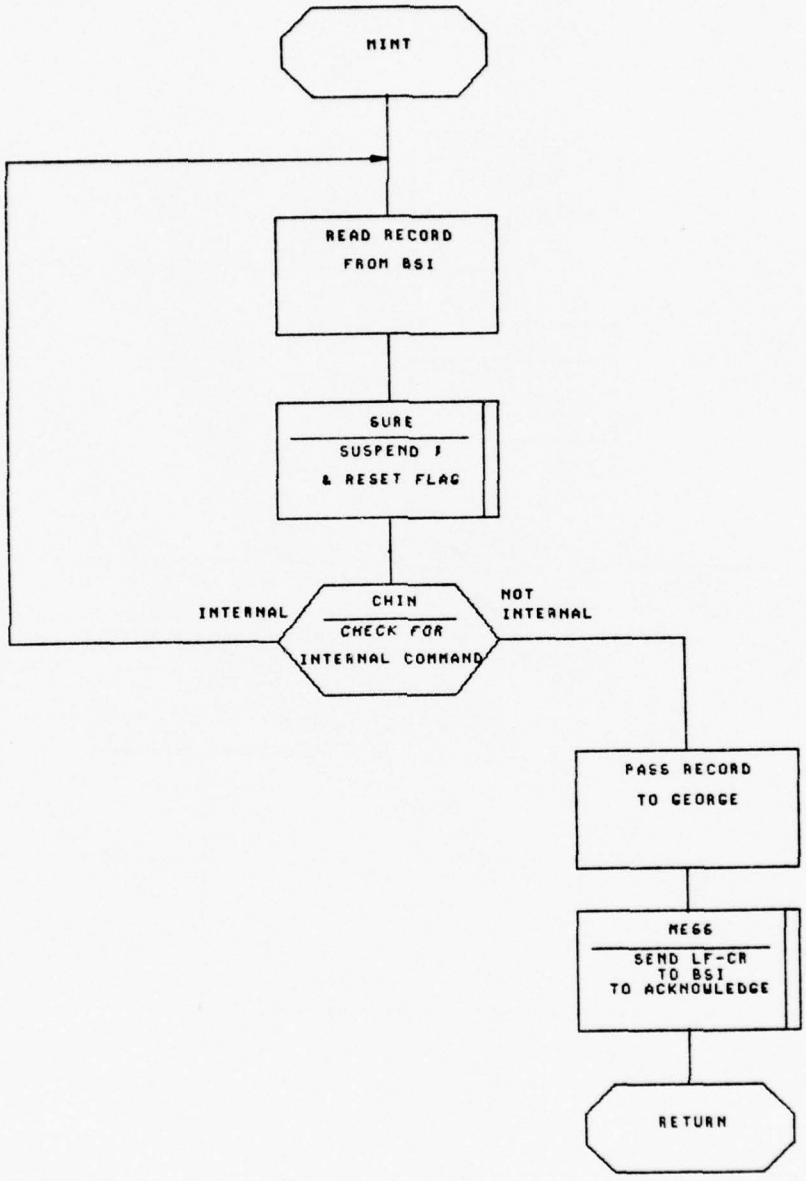


Fig E2 MINT: reads a record from the BSI

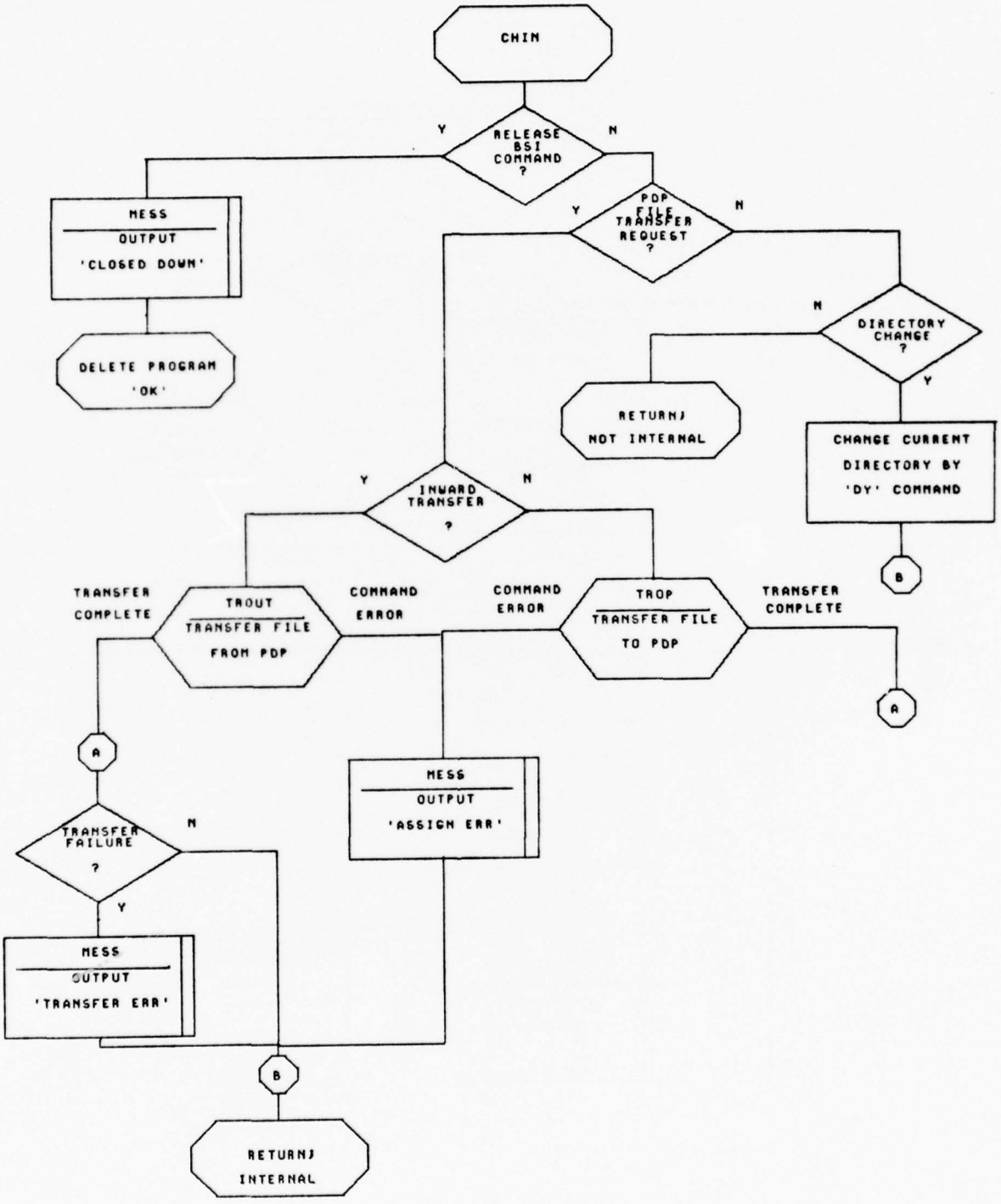


Fig E3 CHIN: checks for internal request

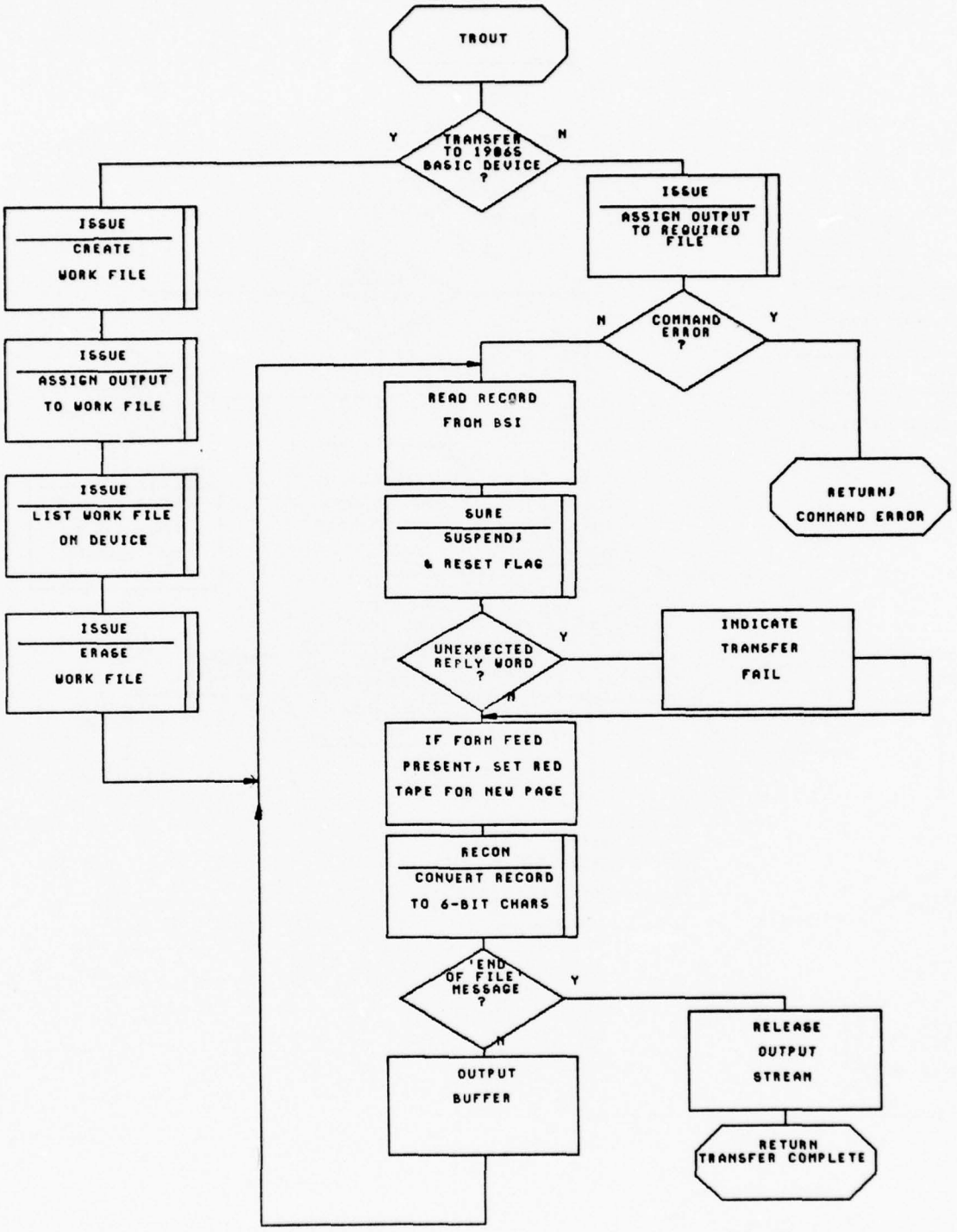


Fig E4 TROUT: transfers a file from the PDP 11

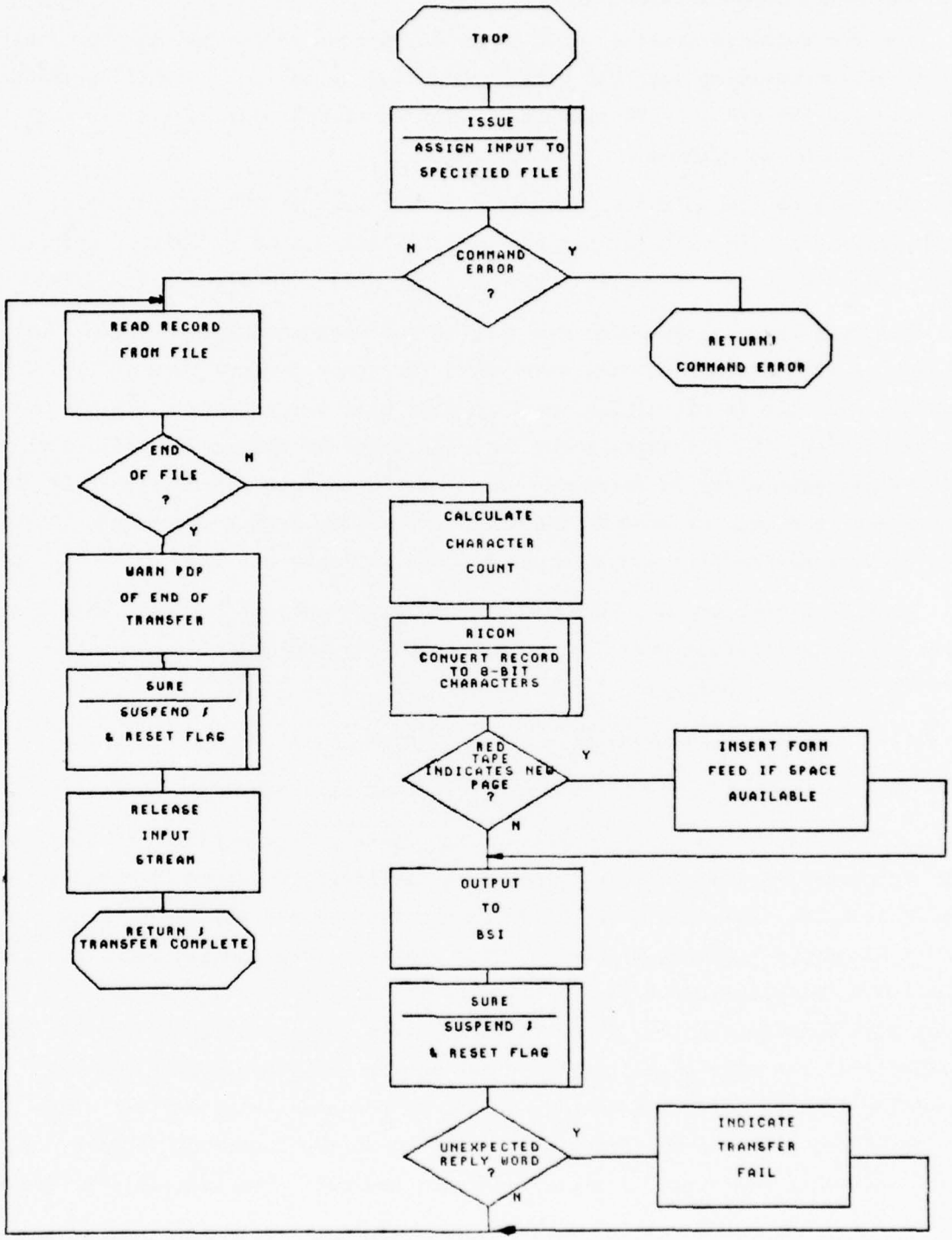


Fig E5 TROP: transfers a file to the PDP 11

E.2 PDP 11 software description

The PDP software consists of a basic BSI driver which handles the interactive MOP conversation together with the routine to implement the file transfers requested via the CYI and CYO commands. Figs E6 to E18 contain the flowcharts for the routines concerned.

The bulk of the software is written in the form of FORTRAN callable, assembler routines so that any new BSI applications can be quickly programmed in a high level language. The main control of the driver is written in FORTRAN.

BSI transfers in the 1906S are carried out autonomously. The user supplies the address of the start of data, the character count and the type of transfer required, and this is executed without further user intervention. In the PDP machine however, BSI transfers occur on a character by character basis, each end of character generating an interrupt which can be used to temporarily halt the execution of the main program in order to service the device concerned. To achieve the transfer of a complete record, the software has to:

(a) provide a routine to be entered whenever an interrupt occurs. This routine checks that the previous transfer was completed successfully and continues to initiate successive character transfers until:

(i) on input, the terminate flag is raised,

(ii) on output, the user's character count expires;

(b) inform the PDP operating system of the address of the interrupt routine. Each device has an associated two word area, at fixed locations in core, which is used to store the address of the interrupt routine and the status required when an interrupt is being processed. The status value, amongst other functions, establishes the priority of the routine, *ie* the extent to which it may be interrupted by requests from other devices. Once the operating system has been provided with the address of the interrupt routine, the routine is entered automatically each time an interrupt occurs on the associated device. The routine NITBSI performs the function of setting up the interrupt vectors for the input (Acceptor) and output (Source) sides of the BSI. The vectors are reset by RELBSI.

In addition to the interrupt vectors associated with each device, there are also two other registers which contain the current status of the device and any associated data. For example, the acceptor status word has bits which indicate the current state of the PDP acceptor operable and acceptor control flags, and the 1906S source operable and source control flags. Conversely, the source status

word indicates the current state of the PDP source operable and source control flags, and the 1906S acceptor operable and acceptor control flags.

TOBSI is the routine which transfers a user record to the BSI. First of all, it checks if the 1906S is asserting 'source operable' and if so it takes an error exit. Otherwise, it adds an ESCAPE character to the output record as a terminator (this is included to maintain compatibility with the Tektronix link), and generates an initial interrupt by setting:

(a) source operable and interrupt enable,

(b) source start.

It then loops waiting for an interrupt to be generated and periodically checking whether the interrupt is being inhibited by the 1906S having set the source operable flag. Once it recognises that the first interrupt has been generated it leaves the work to the source interrupt routine and waits until it receives an indication that the transfer is complete before making an exit.

On entry to the source interrupt routine, the source status register is checked to see if a transfer error has occurred; if so the routine halts. Otherwise the 'source start' flag is cleared to inhibit further interrupts and the 'data ready' flag is cleared to reset the data register. The next character from the output record is moved into the data register, a check is made to ensure that the 1906S acceptor control flag is up, and then the source control flag is set to initiate the transfer. As a further security check the source control flag is not reset until the 1906S acceptor control is lowered. Finally, before returning to resume the interrupted process, the 'source start' flag is again set to indicate that the next interrupt may now be accepted. For the transfer of the last character in the record the 'terminate' flag is raised to indicate to the 1906S that the record transfer is about to be completed.

On the input side of the software, FROBSI is the routine which reads a record of information from the BSI. As an initial precaution it ensures that the 1906S acceptor operable flag and all the PDP source flags are reset. The initial interrupt is generated by setting the PDP acceptor operable, acceptor control and interrupt enable flags. Should an interrupt not be generated, then the routine checks if it is being inhibited by the 1906S having raised its acceptor operable flag. After the first interrupt has been received, successive character transfers are initiated by the interrupt routine until the 1906S sets the 'terminate' flag. FROBSI also handles the Tektronix 'page full' message which is output by the 1906S driver but is of no interest to the PDP user; and establishes whether

GEORGE is ready to receive the next command, *ie* whether the message terminates with 'BACK ARROW,SPACE,BELL'.

The acceptor interrupt routine is very simple. Successive characters received from the BSI are added to the user's buffer and the next transfer is then initiated by setting the PDP acceptor operable, acceptor control and interrupt enable flags. When the 'terminate' bit is asserted by the 1906S, the interrupt routine sets the 'transfer complete' indicator which is polled by FROBSI.

Two other FORTRAN callable routines, TOKB and FROKB handle the teletype input and output. FROKB provides the usual DOS facilities of line deletion (CONTROL/U), character deletion (RUBOUT) and TAB implementation.

File transfers are implemented by the routine TRANS. The PDP file specification is compacted using the standard DOS facility,.CSI1 (command string interpreter macro), is analysed into its input/output constituents using.CSI2 and then the file is opened. Should any errors occur during these procedures then an error exit is taken to output the message :'DEC ERR'. If no 1906S specification has been provided by the user, the routine substitutes the PDP file name minus any device or user identification and with any '.' replaced by '-'. The 1906S specification is then appended to the PDPTRANO or PDPTRANI buffer and the whole is sent to the 1906S driver as a warning that a file transfer is required. If the 1906S driver finds an error in the file specification, then it returns the message 'ASSIGN ERR'. This is either received as the first record of an inward file transfer or its attempted transmission causes a protocol error on the first record of an outward transfer.

For an outward file transfer, the information is read from the PDP file-store, the terminating CARRIAGE RETURN/LINE FEED characters are removed and the record is sent to the BSI. To prevent the transmission if initial control characters other than form feed (these occur in listing files) the first two characters are always examined and any control characters are replaced by spaces. The routine also attempts to insert a form feed character in the first record of the file so that a 1906S listing is produced in a convenient format. When the end of the PDP file is recognised, the 'PDPTRANE' warning is sent to the 1906S as an indication that there are no further file records to be transferred.

When transferring in to the PDP 11, the first record received must be checked against the 'ASSIGN ERR' message. If this is found, the PDP file is closed and deleted, and control is returned to the main routine to take the next

command from the keyboard. Otherwise, a terminating CARRIAGE RETURN/LINE FEED is added to each record received from the 1906S and then is written into the PDP filestore. This continues until the 'PDPTRANE' end of file record is received when the PDP file is closed.

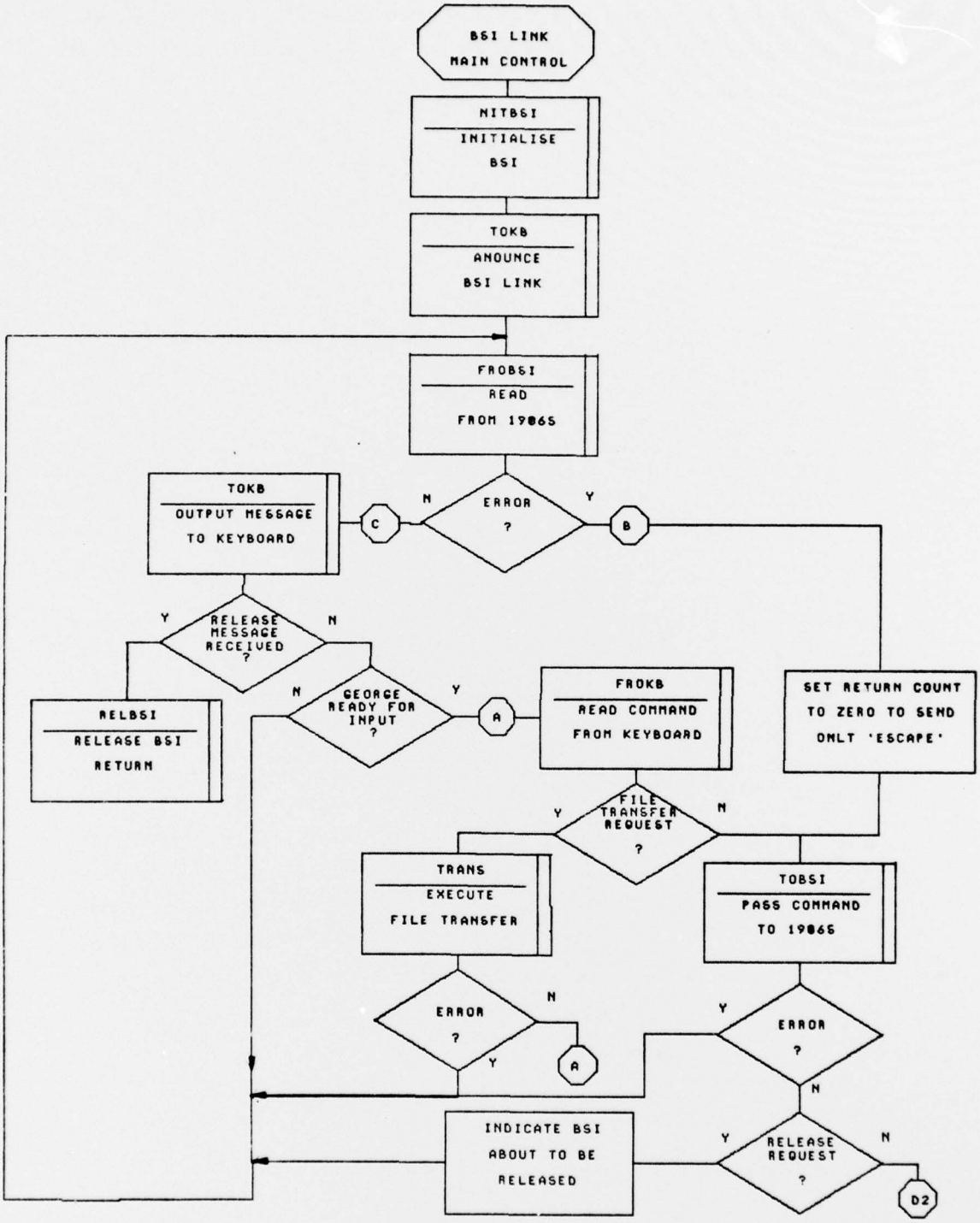


Fig E6a BSI link: PDP main control

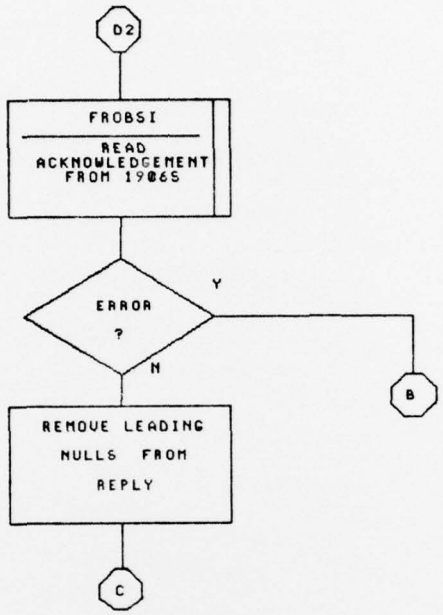
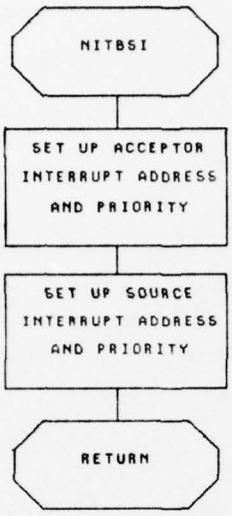


Fig E6b BSI link: PDP main control (continued)



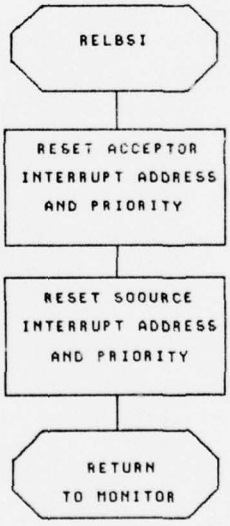


Fig E7 NITBSI: BSI interrupt initilisation
RELBSI: BSI interrupt release

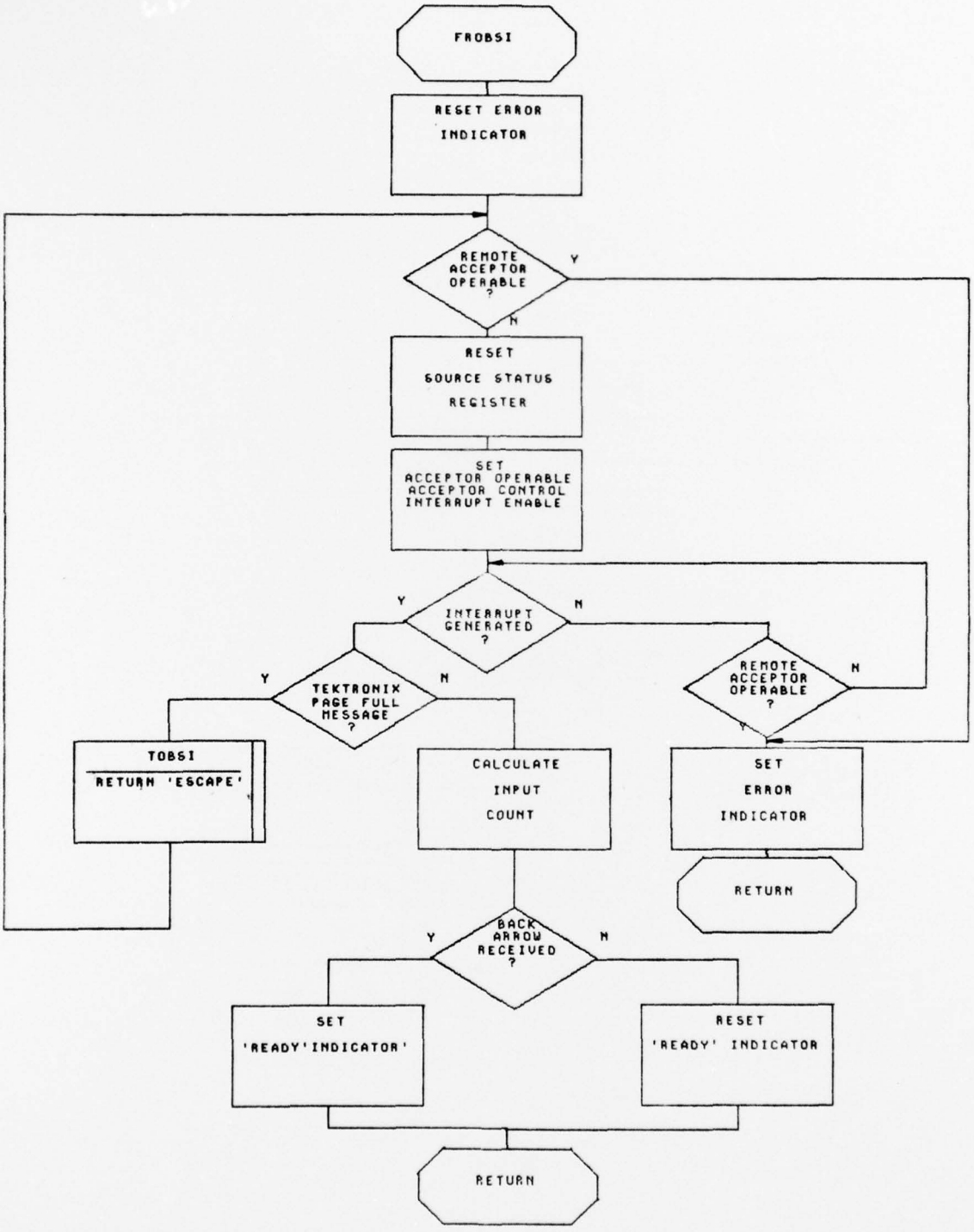


Fig E8 FROBSI: transfers a record from the BSI

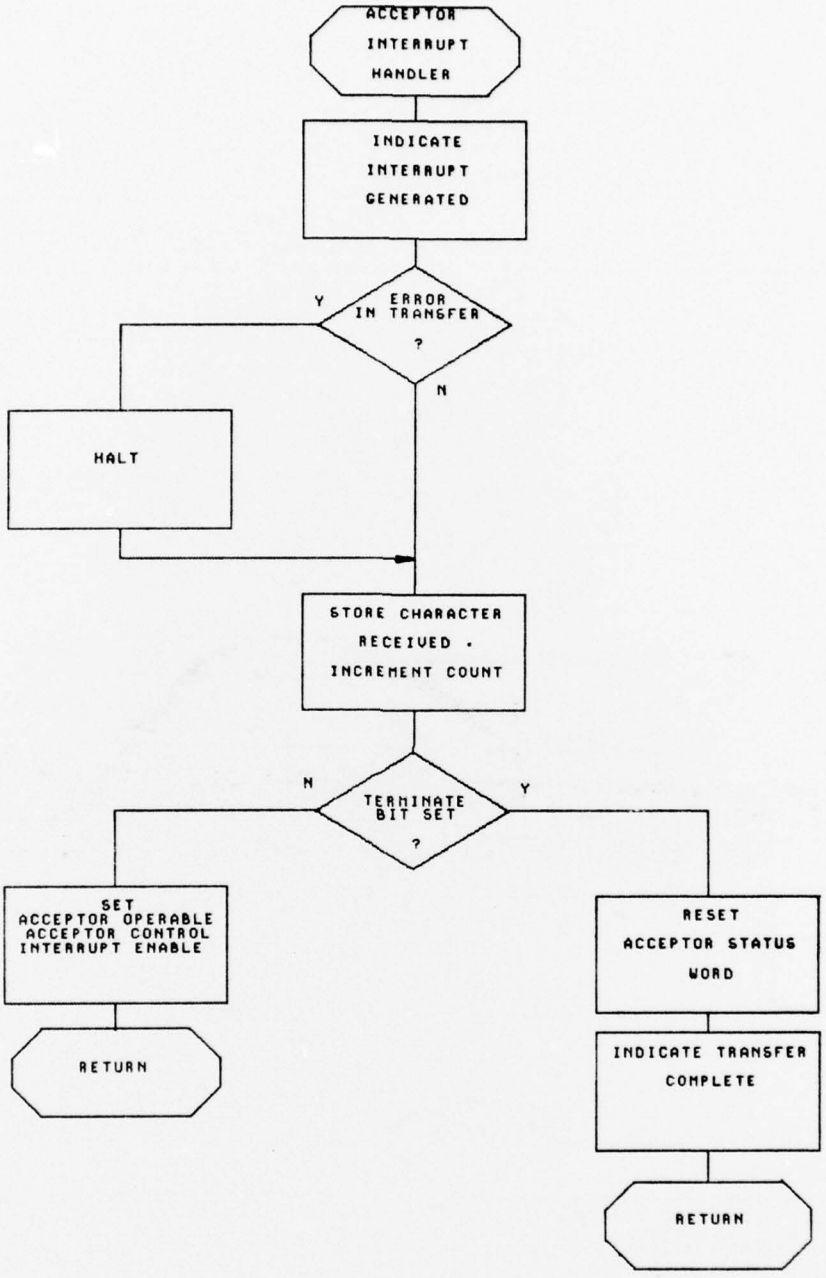


Fig E9 Acceptor interrupt handler

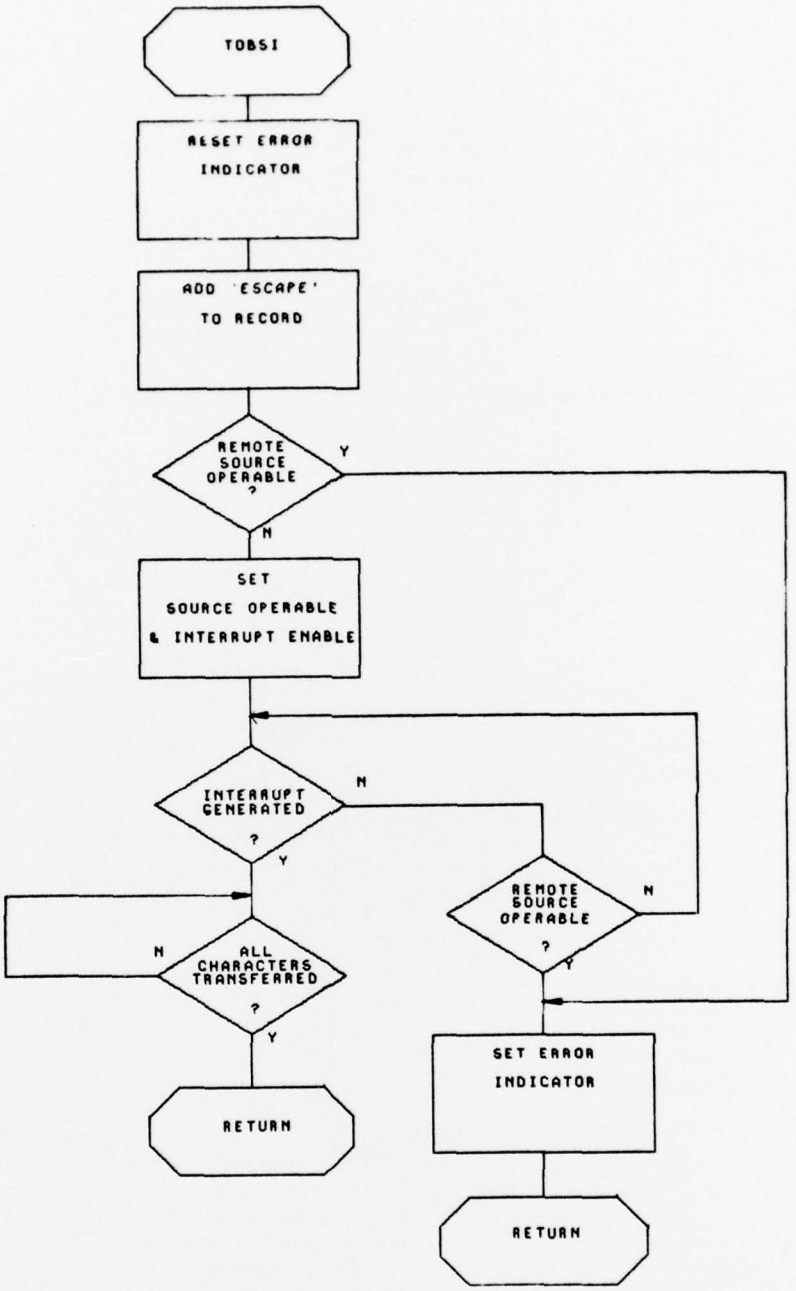


Fig E10 TOBSI: transfers a record to the BSI

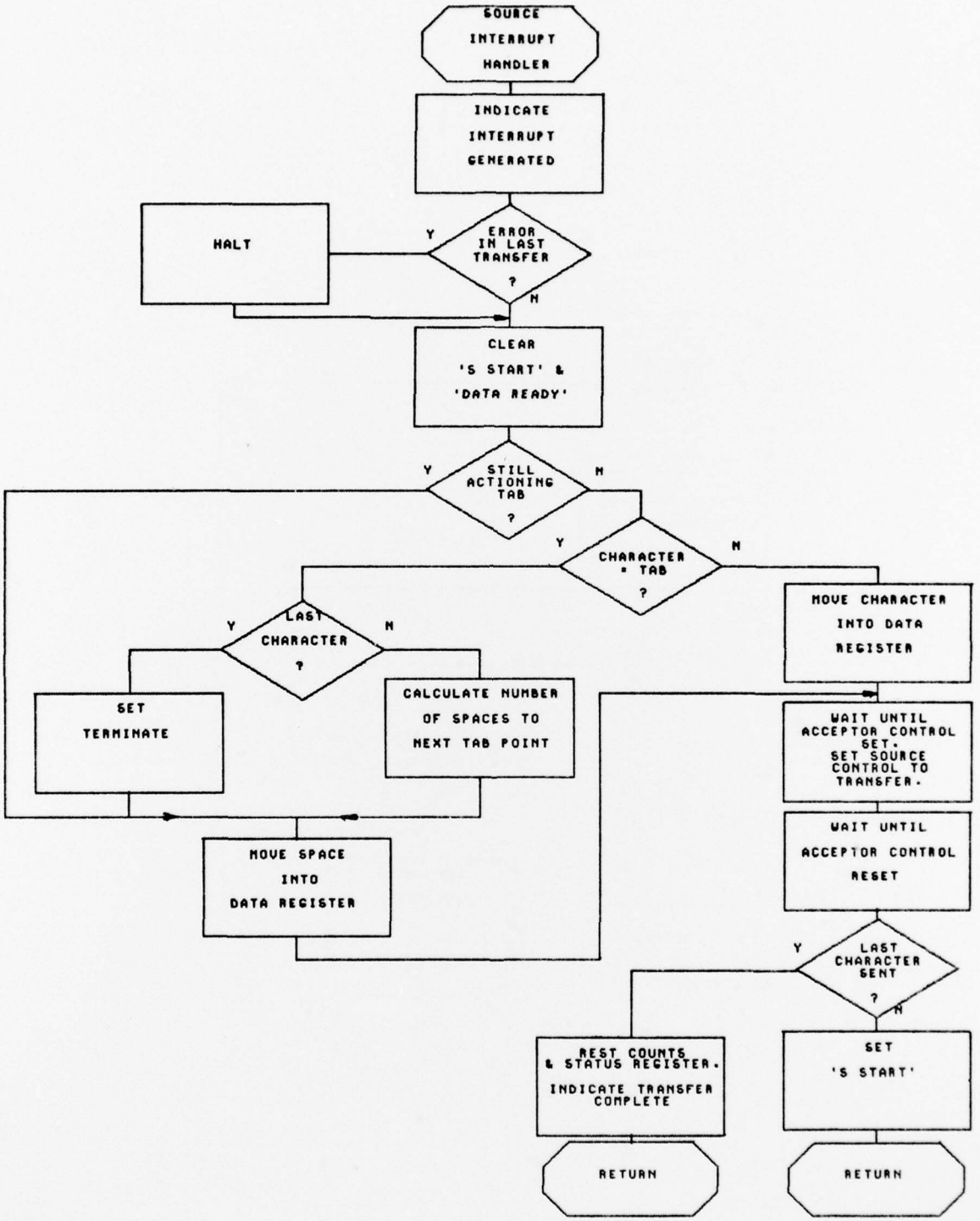


Fig E11 Source interrupt handler

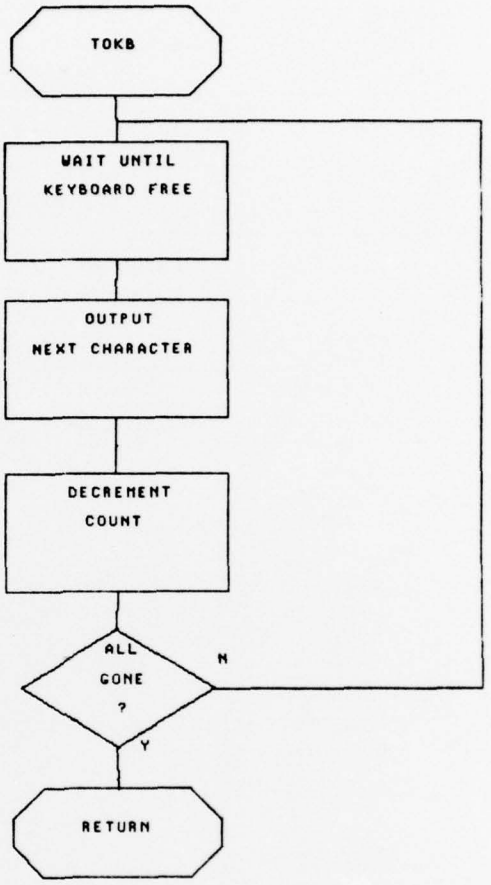


Fig E12 TOKB: sends a message to the teletype

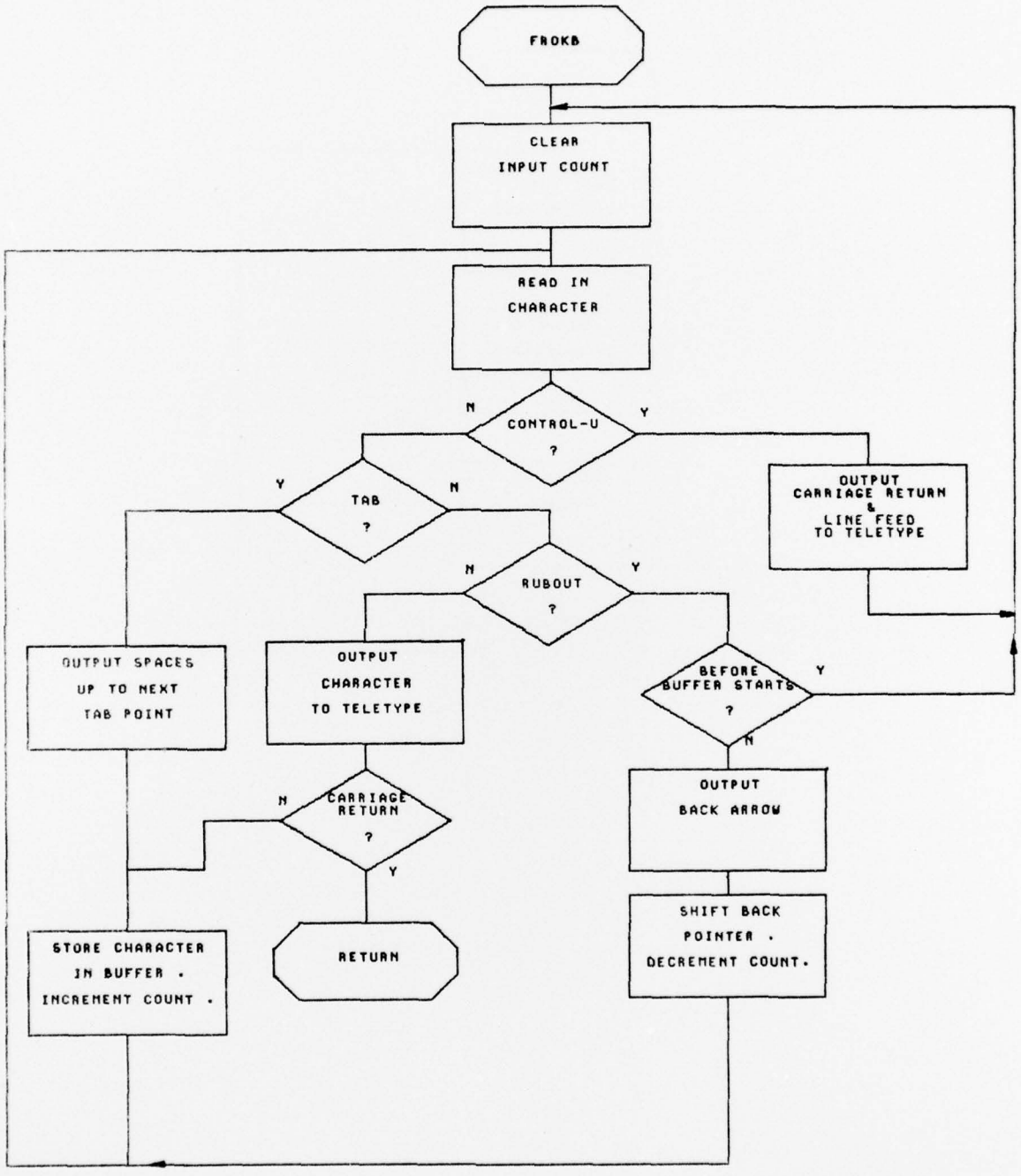


Fig E13 FROKB: reads a command from the teletype

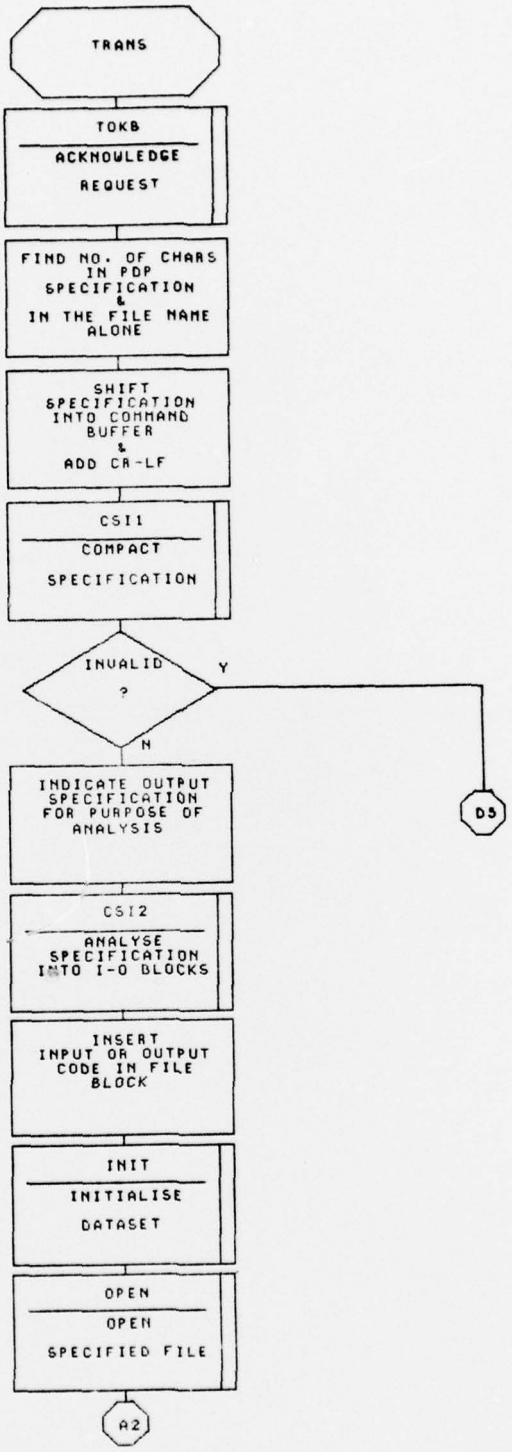


Fig E14 TRANS: *executes a file transfer*
Section 1: analysis of the PDP file specification

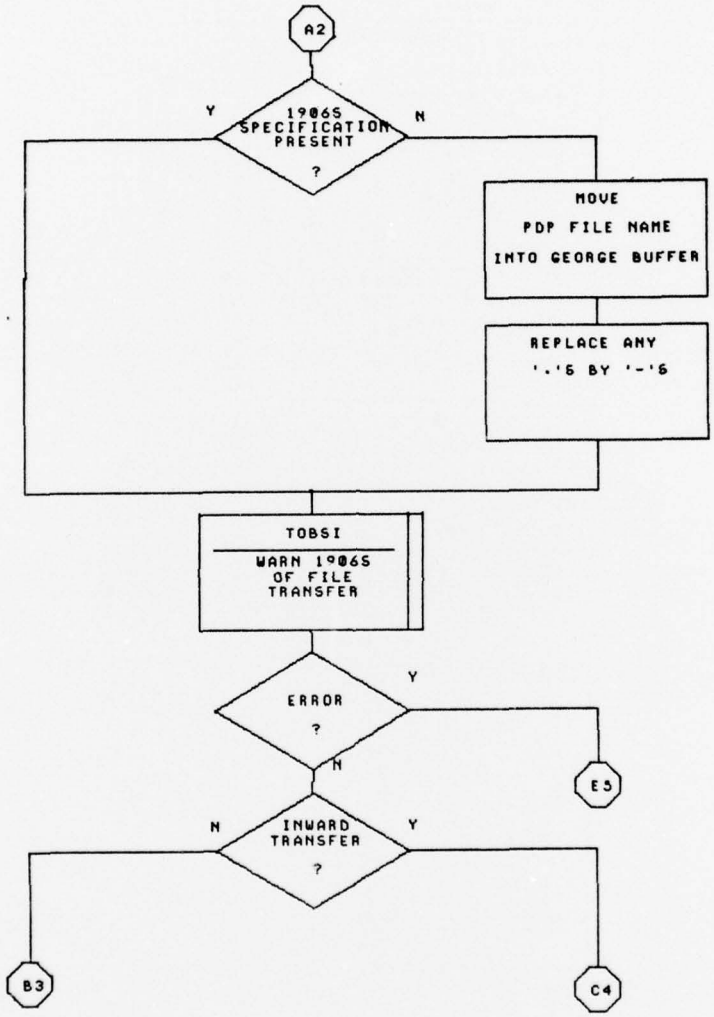


Fig E15 TRANS: executes a file transfer
Section 2: analysis of the George specification

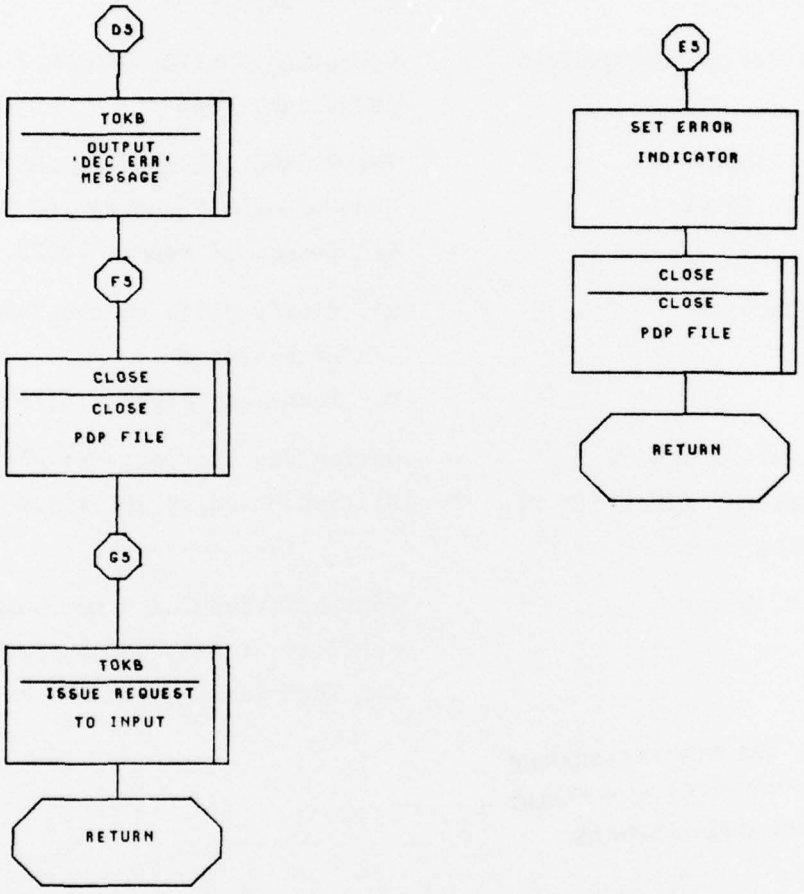


Fig E18 TRANS: executes a file transfer
Section 5: termination conditions

REFERENCES

No.	Author	Title, etc
1	British Standards Institution	A digital input/output interface for data collection system. BS 4421
2	Digital Equipment Corporation	The DOS/BATCH Handbook. DEC-11-ODBHA-A-D
3	International Computers Ltd	Operating Systems GEORGE 3 and 4. Tech. Pub. 4345
4	R.T. Robinson T.R.H. Sizer	The GEORGE 3 Computing Service at RAE Farnborough: Planning and Organisation. RAE Technical Report 76022 (1976)
5	G. Sharples	RAE User's Guide to the TANGO Computer Graphics System. RAE Technical Report 75073 (1975)
6	Digital Equipment Computer Special System Reading	Option Description: BSI PDP 11 to British Standard Interface
7	T.R.H. Sizer	Possibilities for a new Computing Facility at RAE, Farnborough. RAE Technical Memorandum Math 7507 (1975)

REPORT DOCUMENTATION PAGE

Overall security classification of this page

UNCLASSIFIED

As far as possible this page should contain only unclassified information. If it is necessary to enter classified information, the box above must be marked to indicate the classification, e.g. Restricted, Confidential or Secret.

1. DRIC Reference (to be added by DRIC)	2. Originator's Reference	3. Agency Reference	4. Report Security Classification/Marking
	RAE TR 77010	N/A	UNCLASSIFIED

5. DRIC Code for Originator	6. Originator (Corporate Author) Name and Location
850100	Royal Aircraft Establishment, Farnborough, Hants, UK
5a. Sponsoring Agency's Code	**6a. Sponsoring Agency (Contract Authority) Name and Location**
N/A	N/A

7. Title Using British Standard interfaces to create a fast link between two computers

7a. (For Translations) Title in Foreign Language

7b. (For Conference Papers) Title, Place and Date of Conference

8. Author 1. Surname, Initials	9a. Author 2	9b. Authors 3, 4	10. Date	Pages	Refs.
Sharples, G	Drury, G.F.		July 1977	53	7

11. Contract Number	12. Period	13. Project	14. Other Reference Nos.
N/A	N/A		Math 227

15. Distribution statement
(a) Controlled by – [redacted]
(b) Special limitations (if any) – UNLIMITED

16. Descriptors (Keywords) (Descriptors marked * are selected from TEST)

Computer systems programs*. Data links*.

17. Abstract

An ICL 1906S running under the GEORGE 4 operating system and a Digital Equipment PDP 11/40 have been linked using British Standard Interfaces. The PDP 11 user is given access to the 1906S basic peripherals and filestore via the multi-access facilities of GEORGE. He also has the capability of transmitting files between the filestores of the two machines.

This Report describes the implementation of the inter-machine link giving specifications for the hardware and software involved.

F5910/1

RAE Form A143